



Energy, Mines and
Resources Canada

Énergie, Mines et
Ressources Canada

CANMET

Canada Centre
for Mineral
and Energy
Technology

Centre canadien
de la technologie
des minéraux
et de l'énergie

READOUT AND CONTROL SYSTEM FOR A DIRECT READING OPTICAL EMISSION SPECTROMETER

PART II: THE SOFTWARE

T.R. Churchill
Chemical Laboratory

May 1984

Project MRP-3.8.0.1.11
Analytical Methodology

This document is an unedited
interim report prepared primarily
for discussion and internal
reporting purposes. It does not
represent a final expression of
the opinion of the Canada Centre
for Mineral and Energy Technology
(CANMET).

Ce document est un rapport
provisoire non-révisé et rédigé
principalement pour fin de
discussion et de documentation
interne. Il ne représente seule-
ment l'expression définitive de
l'opinion du Centre canadien de
la technologie des minéraux et de
l'énergie (CANMET).

58/4P
MINERALS RESEARCH PROGRAM
MINERAL SCIENCES LABORATORIES
DIVISION REPORT MRP/MSL 84-47 (TR)

SEP 24 1984
MICROMEDIA

READOUT AND CONTROL SYSTEM FOR A DIRECT
READING OPTICAL EMISSION SPECTROMETER
PART II: THE SOFTWARE

by
T.R. Churchill*

ABSTRACT

This report gives a thorough and detailed explanation and documentation of a computer program which is used to control the operation of an Applied Research Laboratories Model 29000/29710 Universal Vacuum Air Quantometer via a Data General Nova 2/10 minicomputer. Commented listings of both the main program, written in the computer language BASIC, and the associated input/output routines, written in "Assembler" language, are provided in the Appendices. The components of the system were described in Part I of this report and Part III will be an operators manual which will provide the detailed instructions for the operation of the system to produce a sample analysis and analytical report.

*Chemist, Optical Emission and NAA Section, Mineral Sciences Laboratories,
CANMET, Energy, Mines and Resources Canada, 555 Booth Street, Ottawa,
Ontario K1A 0G1.

SYSTÈME DE LECTURE ET DE CONTRÔLE POUR UN SPECTROMÈTRE D'ÉMISSION
OPTIQUE À LECTURE DIRECTE

PARTIE II: LE LOGICIEL

par

T.R. Churchill*

RÉSUMÉ

Ce rapport donne une explication approfondie d'un programme informatique qui sert au contrôle du fonctionnement de l'Universal Vacuum Air Quantometer (spectromètre universel d'air et de vide), modèle 29 000/29710 des Applied Research Laboratories par l'intermédiaire d'un mini-ordinateur Nova 2/10 de la Data General. Des listes commentées du programme principal, rédigé en langage BASIC, et des sous-programmes associés d'entrée et de sortie, rédigés en langage "Assembler", sont données en annexe. Les composantes du système ont été décrites à la partie I de ce rapport et la partie III consistera en un guide des opérateurs qui donnera les instructions détaillées pour l'exploitation du système afin de produire l'analyse des échantillons ainsi qu'un rapport d'analyse.

*Chimiste, Section de l'émission optique et de l'analyse par activation neutronique, Laboratoires des sciences minérales, CANMET, Énergie, Mines et Ressources Canada, 555, rue Booth, Ottawa (Ontario) K1A 0G1.

INTRODUCTION

Part I of this report described the stepwise hardware modifications to an Applied Research Laboratories Model 29000/29710 Universal Vacuum Air Quantometer (UVAQ) to arrive at a computer controlled system.

A program was written to provide operator control of the system via the computer and to collect and treat the data obtained. As changes were made to the hardware part of the system it was also necessary to modify the computer program to accommodate these changes. In addition to the changes imposed by the hardware modification, the program was being continually upgraded and new features added as we gained experience in using the system. This report describes the computer program as it existed at the time of writing. It is highly probable that the program will be modified further, but it is expected that the changes will be minor and the major structure of the program will remain in its current form. This report is a rigorous description of the program providing sufficient detail that an experienced programmer will be able to use the report as a guide for maintaining, trouble shooting and modifying the program. Other computer programs for calibration and determination of interelement effect have been written and they are described in separate reports.

The program was written in the computer language "BASIC". There were only two high level languages available for the Data General Nova computer, BASIC and FORTRAN. Since our original computer system could only accept and provide programs via the paper tape reader and punch on the teletype it would have been impractical to use FORTRAN. When cassette magnetic tape and later diskettes (floppy disks) were added to the system it would have been practical to use FORTRAN but it would not have provided any significant advantages over BASIC. The two main disadvantages reported for BASIC are that it is slow and requires a lot of memory. Both of these shortcomings arise because it is an interpretive language. However our application does not require high speed data acquisition capabilities and although the computer system is limited to 64 kilobytes of memory this is not a serious limitation especially with the dual diskettes and the Disk Operating System. The BASIC language offers the advantages of being easy to learn, it is very easy to make program changes (i.e. additions, deletions or corrections) and it facilitates the writing of interactive programs. To provide program control of some of the Input/Output (I/O)

devices such as the Analog to Digital Converter (A/D) the Digital I/O board (DIO) and the I/O subsystem, it was necessary to write some assembly language programs which could be accessed by calls from the main "BASIC" program. The assembly language routines are also described and the call numbers to be used by the main program, the parameters to be supplied and the function of each routine is listed in Table 1.

In writing the operating program the two major objectives which dictated the design of the program were to use a kernel or kernels of basic, fixed information which would be used to direct the operation of the instrument and to derive the final results, and to make the program interactive (actions implemented via the keyboard) and easy for the operator to use. Both of these objectives have been realized in the present program and implementation of the interactive feature was facilitated with the addition of the console keyboard which provides for up to 44 user functions to be selected via function keys. Basically what the program must accomplish is to permit the selection of elements to be analysed for (i.e. selection of spectrometer channels to be read), the selection of the basic fixed information which depends on the matrix type of the samples to be analyzed (i.e. low alloy steel, high alloy steel, cast iron, cupro-nickel alloy etc.), the collection of the data and conversion of the raw data to concentration values for each of the elements selected. In addition to the basic functions of the program a number of auxiliary functions are available which allow the operator to change the list of elements selected, to delete the data for selected analyses, to direct the output to the printer or to store it on diskette; in all there are presently 12 functions which can be selected by the operator via the function keys.

MAIN PROGRAM OUTLINE

A general outline of the main program is shown in the flowchart of Figure 1. The kernel of basic information required to perform a sample analysis is placed in a table or two dimensional array. Since this information is different for each matrix or type of alloy to be analysed, it is stored in diskette files. When the program is started the operator

is requested to select the appropriate file and the information is entered into the S array. Then the operator has the choice of retrieving a list of previously determined standardization coefficients and having them placed in the B vector or leaving all the coefficients at a value of 1. If the B coefficients are left at the value 1, then the next function to be selected should be the standardization routine. When the standardization routine is requested the operator must do a minimum of three analyses of a secondary standard sample and this data is used to determine a new set of coefficients which are then placed in the B array and also saved in a diskette file named Coeffs. If the low alloy steel matrix has been selected the next option the operator can choose is the particular elements which he wishes to analyze for. When this selection is made the program creates a singly-linked list which is then used to select the data pertaining to the elements selected. The program then displays the symbol for each element selected and requests a sample identification number or description. The program is now set at the common return point from all routines (input point 3) and is awaiting keyboard input. If a function key and then the carriage return (CR) key is pressed the program will branch to the routine corresponding to the function key. If only the CR key is depressed or some other non-function keys and then the CR key the program goes to the Data Collection Routine or analysis mode. At the completion of one analysis or sparking of the sample (which is controlled by the program) the program displays the adjusted reading for each element selected and returns to entry point 3. The operator now has the option to select another analysis of the same sample or any of the available functions. Since keyboard input responses can be stacked the operator could key in a sequence of operations and then he would only have to ensure that the sample is correctly positioned after each analysis.

MAIN PROGRAM DETAILED DESCRIPTION

Appendix A contains a commented listing of the main program and Appendix B lists the variables, string variables and arrays, their function and the program line number where they are used.

The fixed data or information required to perform an analysis is contained in the string variable MD\$ and the two dimensional array S. The string MD\$ contains 3 letter codes for each of the elements for which the instrument has an analytical measuring channel. For some elements there is

more than one physical channel or more than one set of calibration data, hence the 3 letter codes contain the chemical symbol for the element and if that particular element has more than one physical channel or set of data a number suffix is added. For example aluminum has three codes in the MD\$ string AL, AL1 and AL2. The 2 dimensional S array contains up to 14 pieces of data for each element entry. Each row of the array contains data for one element. Figure 2 indicates the meaning of each column in the S array.

As mentioned previously the data for the S array is read into the computer from a diskette file. The routine to accomplish this task and also read in values to the B vector if requested begins at line 310 of the program. The data for the various alloy types are stored in files which have filenames which differ only in the two character filename extension (i.e. ALLOYTYPE.01, ALLOYTYPE.02 etc.). The set of extension characters are stored in string variable C1\$ and the operator response number is used to select a pair of extension characters and add them to the filename which is stored in string variable F1\$ (line 330). The data is then transferred from the diskette file into the computer. Next the operator can choose to enter previously determined coefficients or not. If the old coefficients are to be read in, this is accomplished by the subroutine commencing at line 4870. The data is first transferred into the B vector and then into column 14 of the appropriate rows of the S array. It will be noticed that the first coefficient transferred from the B vector is from the second location. This arises because the first coefficient is associated with the internal standard or main matrix element which does not have any associated data in the S array.

At this stage of the program all the fixed data required to do an analysis has been entered into the computer memory. Next the elements to be analysed for are selected. If the alloy type is not low alloy steel the complement of elements is automatically selected otherwise the operator can select a standard set, a standard set plus additional elements or any selection of elements for the low alloy steel. At this point the operator has the option to go to the standardization mode prior to element selection, because in the special case of low alloy steel the standardization is applied to all possible elements even though only a selected set may be analysed for. If at a later time the operator wishes to add other elements or different concentration ranges of the same element he does not have to repeat the standardization procedure.

The standardization procedure produces a set of coefficients which are applied to the data to compensate for day to day changes which can occur in the data output of the spectrometer. This can arise because of such things as changes or fluctuations of line voltage, slight changes in the output of the source unit, and the build-up of a film on mirrors or lenses. The coefficients are determined by making measurements on a secondary standard which was measured at the time the original calibrations were made and taking the ratio of the original measurements to the current measurements. The original measurement values are stored in the S array in column 7 and the ratios or coefficients are stored in column 14 and the B vector. The routine which performs this task begins at line 4460. Variable F7 is set to 2 which indicates that standardization is in progress. If the standardization is for low alloy steel then all possible elements and channels are linked (lines 4467 to 4475). Otherwise the linkage has already taken place and the program proceeds to line 4477 which sets the values in the B vector to 1. The program then proceeds to the data collection routine at entry point 2 (Figure 1). The F7 variable is checked and since it is set at 2 a flashing message is displayed on the video terminal to advise the operator to analyze the secondary standard or check sample (line 890). When the measurements on the secondary standard have been completed the results are averaged (by operator action). The average routine checks to insure that 3 or more measurements have been made, calculates the average and then returns the program to the standardization routine at line 4510. The routine stores the coefficients in the correct rows by making use of the link values in the 1st column of the S array. Line 4510 assigns the value of the top of list pointer to variable T1, this value is then used to retrieve the values from column 7 to determine the coefficient (line 4540) and to store the resulting value in column 14 of the same row. Line 4560 locates the next item in the linked list. When all coefficients have been calculated and stored they are listed on the printer to provide a permanent copy for the operator and also stored on diskette (lines 4600 to 4740). The routine has two possible return routes depending on the matrix selection (i.e. the value of A1). If low alloy steel was selected then the operator has the option to choose a selected group of elements otherwise the return is to entry point 3.

DATA COLLECTION ROUTINE

The data collected for each channel of the selected group of channels or elements are placed in vector D. The D vector is dimensioned to hold 140 data items. The first N locations of the D vector (where N is the number of elements being measured) contains the cumulative total of multiple measurements on one sample. The cumulative total is used to determine the average value for each element. The data for successive burns on the same sample are placed in succeeding groups of N locations for each analysis (see Figure 3). In order to keep track of the groups of N locations which are occupied or available another vector F is used. Prior to a series of analyses on a sample the D vector and F vector are cleared (all values set to 0), then each time an analysis is performed and the data placed in the D vector a corresponding location in the F vector is set to 1. If the operator chooses to delete the data for a given measurement the corresponding locations in the D and F vector locations are set to 0 after the values in the first N locations of the D vector (the cumulative totals) have been adjusted. At the completion of one data collection sequence the program returns to entry point 3 (see Figure 1) and the operator can choose to do more analyses or select any of the available functions.

The data collection routine begins at line 1030 of the program where the first action is to check the F vector to locate the first available area in the D vector. If the D vector is filled then a flashing message "BUFFER FULL" is displayed on the video terminal and the program returns to entry point 3 to allow the operator to take appropriate action. If a data area is available the index to the first available location is calculated at line 1040 and a check is made to ensure that sufficient locations are available in the D vector. The program then branches to a subroutine which monitors the internal standard channel of the spectrometer, which controls the collections of data in a set of integrators, and stops the analysis when the internal standard channel has reached a pre-set value. On return from this subroutine the voltages accumulated in the integrators are converted to digital numbers via calls to assembly language routines (lines 1055 to 1095). In the current version of the program each integrator is read 20 times (loop beginning at line 1055) and the cumulative data are averaged to compensate for noise in the analog to digital converter. The digital numbers are placed in the D vector and then the values are adjusted by applying correction factors and displayed on the video terminal

(lines 1100 to 1155). The integrators are next reset to 0 in preparation for the next analysis (lines 1160 to 1170), the appropriate location in the F vector is set (line 1190) and the program returns to entry point 3.

ROUTINES ACCESSED BY USER FUNCTION KEYS

Table 2 lists the routines which can be selected by the operator by the use of a function key. Except for two keys on the screen management pad of the keyboard (see Figure 4) the user function keys are the top row of keys. Each key can provide four different codes by using it alone or simultaneously with either the Shift or Control (CTRL) or simultaneously with both the Shift and CTRL keys. The routines are identified by 3 letter mnemonics and these are printed on a label directly above the corresponding key. Most of the function keys produce codes which are equivalent to a printable character and these characters (see Table 2) appear on the video screen when a function key is pressed. The program when operating will eventually arrive at entry point 3 (Fig. 1) at line 930 where it is awaiting input from the keyboard (i.e. operator action). A carriage return (CR) is considered as an end of keyboard input signal and if only the CR key is pressed there will not be any characters entered into the A2\$ string variable and the program branches immediately to the data collection routine from line 935. If the first key pressed was a function key then the first byte in A2\$ will contain a code equivalent to decimal 30. Line 940 checks to see if the first key pressed was a function key and if it was not then the program branches to the data collection routine regardless of any other keys pressed. The operator could type in notes or messages about the analysis to follow and these will appear on the screen. If the first key pressed was a function key then the program continues to line 950 which determines which key was pressed. The string variable B0\$ contains the codes which would be generated by the top row of function keys plus two others. At line 950 the variable C1 will contain the ordinal position in B0\$ of the code contained in the second byte of A2\$ (i.e. A2\$(2,2)) the value of C1 is then used to branch to the beginning of the associated routine at line 970. If a function key which does not have an associated routine is depressed the program will branch to line 1010 and display the message "Routine Not Available" and return to entry point 3 at line 930.

In the paragraphs which follow the routines which are invoked by the function keys are described in the same order as their codes appear in the string variable BO\$ which is in order from left to right of the top row of function keys (see Figure 4).

ROUTINE TO DETERMINE AVERAGES

This routine is used to determine both the average and the relative standard deviation and thus is entered when either the key identified by "AVG" or "RSD" is pressed. The routine also branches to a subroutine which converts the average values to element concentrations except in the special case of standardization. First the variable F8 is set to 1 to indicate that an average has been calculated, this variable is reset to zero when the current data is erased. The F8 variable is used by the print routines to verify that average values are available. Lines 1500 to 1520 determine the number of analyses or burns by checking the number of locations in the F1 vector which have been set and places this number in variable J2. Line 1530 is used to check if the data is being used for standardization to ensure that at least 3 analyses have been performed. If the two conditions are not satisfied the program branches to line 890 and displays a message on the video terminal to perform 3 or more burns on the sample and then ends up at input point 3. Lines 1550 and 1555 determine the average of the data and the sum of squares for the internal standard and lines 1560 to 1610 calculate the averages for the other elements and prints the results on the video terminal and stores them in the C and A vectors. At line 1620 the program branches to line 3360 if the RSD key was depressed. Line 1630 is a check to determine if the average is to be used in the standardization routine, if not the program branches immediately to the concentration calculations.

Subroutine to Calculate Concentrations

Two methods are used for the calculation of concentrations; one method is used for low alloy steels and cast irons and both methods are used for high alloy materials. The routine for the first method begins at line 2940. Since data from the S array will be used the pointer to the top of the linked list is assigned to variable T. In the loop beginning at line 2980 the average values which are stored in the C vector are first compared with maximum and minimum values in the S array and if either condition is

found to be true then the corresponding maximum or minimum concentration value is displayed (lines 3090 and 3120). If the average value is within the accepted range then the concentration is calculated (lines 3010 to 3060). Since the coefficients relating element concentration to the analysis values could represent from a first order up to a fourth order least squares fit an iterative method is used to solve the equation. The first estimate (line 3010) uses the solution to a first order fit and then this value is inserted in a fourth order equation and tested to determine if the estimated value gives a reading within 0.2 per cent of the actual reading. If the test is true the concentration value is displayed and stored in the C vector. If the test is false another estimate is calculated (line 3050) and tested. It has been determined by numerous tests that even for the worst case of a fourth order fit the maximum number of iterations is less than ten. Of course if the original fit was first order the solution is immediately determined by the first estimate. The routine applies a correction to the silicon values if molybdenum was also present in the samples (lines 3210 to 3260) and again displays all the concentration values, otherwise it returns to the line after the original subroutine call. On return from the subroutine a check is made to determine if the samples were high alloy materials (lines 1650 and 1660) and if either test is true the program branches to a routine which applies corrections to the originally determined concentrations by a technique referred to as the concentration ratio method. The original concentrations values are preserved in the R vector. At the completion of either of the above two concentration calculations the program returns to the common return point 3 at line 930.

ROUTINE TO ERASE DATA

This routine is entered when the second function key is depressed and it clears a number of vectors and displays a heading at the top of the screen in preparation for the analysis of another sample. The routine begins at line 640 and from there to line 770 two variables, four vectors and the video display screen are cleared. The routine then branches to a subroutine to calculate values which are used to determine the day and month. On return from the subroutine the date and time are displayed, a list of the elements to be determined (via subroutine at 4800) and either a message to analyze the check sample in the case of standardisation or a

request for the operator to enter sample identification information (line 910). The sample identification is stored in string vector I1\$ and the program awaits keyboard input at line 930.

Subroutine to Calculate Date Constants

Three letter abbreviations for the days of the week are contained in string variable D1\$ and for the months in string variable M1\$. This subroutine calculates constants which are then used to select the correct three letter abbreviations from the two string variables. Sys(1), Sys(2) and Sys(3) are system variables which return the day of the month (1 to 31), the month of the year (1 to 12) and the year.

ROUTINE TO CALCULATE PER CENT RELATIVE STANDARD DEVIATION

This routine has the same entry point as the average routine at line 1480, since averages of the original data and the squares of the original data are required. At line 1620 of the average routine a branch to the routine is taken if the RSD or third function key has been pressed. The calculation starts at line 3380 after a heading has been displayed and the flag variable F9 is set to 1. A short form method for calculation of the standard deviation is used, that is the square root of the difference between the mean of the squares and the square of the mean. This value is modified to give an unbiased estimate and converted to per cent relative standard deviation (line 3390). Each calculation is displayed on the video and the routine then returns to the average routine for the concentration calculations.

ROUTINE TO DELETE LINES OF DATA

This routine first requests the operator to enter a line number but it will also accept a function key code. It is for this latter reason that a string variable A2\$ is used to accept the input rather than a numeric variable. If it was not a function key then the number entered is converted to a numeric value and assigned to variable W1 (line 1692). If the number is greater than ten it is not valid and the program returns to 1680 to request another line number. Line 1700 resets the flag corresponding to the line number to zero to indicate the data area is available. The index to the area in the D vector where the data is stored is next calculated (line 1710) and then the loop (lines 1720 to 1760)

adjusts the cumulative totals and clears the data area. The routine then returns to the start point and requests another line number. The operator has the choice to enter another line number, press a function key or provide a nul response by pressing the carriage return which will return the program to input point 3.

ROUTINE TO ADD ELEMENT SYMBOLS TO CURRENT LIST

The routine begins at line 560 where a request for the operator to enter the element symbols is displayed. The three character symbols must match one of the set of symbols stored in the string vector MO\$ and if multiple symbols are entered they must be separated by one character (any character). The routine then branches to a subroutine which re-arranges the current linked list to accommodate the additional entries. On return from the subroutine the operator may make further additions or deletion of symbols (line 620) by pressing the appropriate function key. If only the CR key is pressed the program continues to the Erase routine which was described earlier.

Subroutine to Insert Entries into Linked List

This routine implements an algorithm given by Knuth¹ for the insertion of an item into a singly linked list. The position of the rows of data in the S array corresponds to the ordinal position of the symbols in the string vector MO\$. The loop beginning at line 2485 steps through the list of symbols entered by the operator and adjusts the link values in the first column of the appropriate entries of the S array. The link value is in fact the row index of the row of data in the S array corresponding to the next element in the linked list. At line 2490 the "POS" function is used to find a match between three characters from the string variable L2\$ (which contains the symbols entered by the operator) and three from the master list of symbols in MO\$. If a match was not found then the variable P3 will be set to zero, otherwise P3 will contain the byte position in MO\$ of the first byte of the matching string. The ordinal position of the matching symbol in MO\$ is then calculated (line 2510) and assigned to variable J. The linked list is then searched to determine if the new item should be placed at the beginning of the list, in the middle of the list or if it is already entered in the list. Once the position of the new item in the linked list has been determined then the link value in the preceding

position must be changed to point to the new item and the link value (i.e. column 1) of the new item is set to the row position of the succeeding item. Notice that if the new item is the first item in the list then it is the pointer variable P which is changed to point to the new item (line 2710). Each time a new item is added to the linked list the variable N is incremented to indicate the increased number of elements. In the special case of the standardization procedure, the element symbols appear in the string variable L2\$ in the same order as they appear in MD\$ and therefore the search starting position in MD\$ can be advanced by adjusting the variable N4. This adjustment of N4 prevents the POS function from starting at the beginning of MD\$ each time a new search is initiated. At the completion of adding new items to the linked list the program branches to a subroutine (line 2740) which adjusts some variables to accommodate the new linked list.

Subroutine to Set Variables for a New Linked List

Each time a linked list is formed or changed by addition or deletion of elements then the string variable LO\$ and the vector A must be adjusted. The string variable LO\$ contains the element symbols in a format for displaying or printing these symbols as a heading when listing the results of an analysis. The A vector contains the physical channel numbers and the gain settings (0,1,2,3) for each element in the linked list. The subroutine traverses the linked list and uses the link values, assigned to variable P1 to form the formatted element symbols (line 2800) and to extract the corresponding channel numbers and gains from the S array (lines 2810 and 2820). The appropriate B coefficient is also set by this subroutine (line 2835). The call at the end of the subroutine transfers the data from the A vector to a buffer in the assembly routine which is used to control the operation of the analog to digital convertor.

ROUTINE TO REMOVE ELEMENTS FROM LINKED LIST

This routine is analogous to the previous routine and uses the same principles. The operator enters the element symbols for the elements to be removed from the linked list (line 1780) and the routine traverses the linked list searching for matching symbols. When a match is found the link value of the preceding item in the list is adjusted to effectively eliminate the item which is to be removed. When all requested items have

been removed the routine branches to the subroutine to make the required modifications. Notice that if the operator enters a symbol which does not correspond to an entry in the current linked list it is ignored (line 1940), however, an incorrect symbol will cause an error message to be displayed (line 1810).

ROUTINE TO PRINT ANALYSIS DATA

All the data resulting from an analysis are displayed to the operator on the video display and are also stored in various arrays until they are erased by operator action. This routine identified by the symbol PRT provides a hard copy of the data, excluding the original or raw data, via the printer. At the beginning of the routine the operator is provided with four options to choose from which will determine the information to be printed (lines 3500 to 3535). Since the average values are always printed the first line of the routine (line 3470) checks a flag (variable F8) which indicates if non-zero, that the average has been calculated. If the average has not been calculated, a message is displayed and the program branches to input point 3 (line 930) to permit the operator to select the average routine. If a printout of the relative standard deviation has also been requested a similar check is made on variable F9 (lines 3550 to 3510). If the operator selects options 1 or 2 then a heading will be printed which includes the date and time and the list of element symbols corresponding to the date (lines 3600 and 3610). The subroutines to calculate the date constants have been described and the subroutine to print the heading uses the date constants and the formatted symbols in string variable LO\$. This latter subroutine begins at line 2280 and merely ensures that for a long list of elements a carriage return is inserted to avoid overprinting at the end of a line. After the heading is printed or in the case where a heading was not requested the routine next prints the per cent of iron present if a high alloy material was analysed (line 3720) otherwise a sub-heading "RSD" is printed and then the actual values if this option was selected (lines 3750 to 3830) followed by the sub-heading AVG (#) and the values (lines 3840 to 3920), then "RATIO" and the values if the material was high alloy (lines 3940 to 4020) and finally the sub-heading "CONC" and the concentration values (lines 4030 to 4260).

ROUTINE TO RESTORE ANALYSIS READINGS TO DISPLAY

Often one or more lines of information are lost to the display when the screen becomes full and the addition of further lines causes rollover (i.e. each line is shifted up one and the top line is lost). This routine permits the operator to restore all the readings, which have been acquired for the current analysis, to the video display.

The routine first branches to a subroutine at line 4800 which displays a heading consisting of the element symbols stored in string variable L0\$. The F vector is then searched to determine which areas in the D vector contain data (loop at lines 1380 to 1400) and when an occupied data area is located the index into the D vector is calculated (line 1420) and the data is displayed with an identifying number (lines 1425 to 1450). When the F vector has been completely searched the routine returns to the main program return point at line 930.

ROUTINE TO STORE SECONDARY STANDARD READINGS

The output of the spectrometer is checked frequently during the course of a series of analyses by measuring a secondary standard sample (also used for standardization). If these measurements indicate a change in the output has occurred then a correction factor should be applied to subsequent readings. This routine accomplishes the above task when the operator presses the ninth function key after making measurements on the secondary standard. The readings for the element Carbon are used to test if a significant change has occurred. First a check is made to determine if Carbon is the first or second item in the linked list and the variable Z1 is set to the correct value (lines 5231 and 5232). If the average reading for Carbon has changed by more than 1.5% of the standard value (stored in S(3,7) of the S array) then the new readings for all the elements in the current linked list are stored in column 6 of the S array and the ratio of these values to the standardized values are listed on the printer as factors. All subsequent readings will be modified by these factors (line 1110 of the Data Collection Routine). The routine steps through the linked list and stores the new readings (uncorrected by any previous factors) and prints the ratios (lines 5240 to 5270).

STANDARDIZATION ROUTINE

This routine has already been described under the descriptions of the main program since there is an option to select standardization near the beginning of the program. This routine can also be accessed by a function key on the screen management pad (marked STD) so that during the course of daily operation the operator has the option to repeat the standardization procedure.

ROUTINE TO PRINT CURRENT ANALYSIS DATA

This routine is analogous to the routine to restore analysis readings to the display. The routine provides a hard copy of the data together with the current date and time. The routine is initiated by simultaneously pressing the "CTRL" key and function key eight (8). The routine starts at line 5000 and first prints the date and time and then a heading identifying the elements. The F vector is searched (lines 5020 to 5040) and for each non-zero value the program branches to line 5060 where an index to the D array (variable LO) is calculated from the loop variable M. The printer is then activated (line 5070) and the data are printed (lines 5080 to 5150). This process continues until the F vector has been completely searched and then returns to the common entry point at line 930.

ROUTINE TO PRINT FORMATTED ANALYSIS REPORT

The report which is sent to the customer who requested the analysis is produced by this routine and it provides the concentration values for each sample submitted, the laboratory identification number assigned to each sample, the customers identification number, a report number and the project number. A sample of a typical analysis report is shown in Figure 5.

This routine could eventually be developed into a more complex and flexible separate program and for this reason the variable declarations are made within the routine. In its present form it uses several variables and a function routine from the main program and therefore the routine must be used in conjunction with the main program. After the variable declarations (lines 5340 and 5350) the string variable NO\$ is assigned a list of names each of which must contain 15 characters (any unoccupied bytes must contain the code for a space). The names are those of the people who most frequently require copies of an analysis report. The list can be easily

modified by changing one or more of the assignment statements (lines 5360 to 5400). Next the variable N2 is assigned the number of elements determined during the analysis (including the internal standard) and the list of elements is displayed on the video terminal (lines 5415 to 5440). The operator is now given the opportunity to remove data for selected elements (line 5450) and if the response is "YES" the program branches to the subroutine beginning at line 6300. In the subroutine the position number of the element to be deleted is requested (this refers to the position of the element in the list which is displayed on the video terminal) and this number is placed in the F array and a counter (variable LO) is updated (lines 6310 to 6370). This process is repeated until the operator gives a nul response (i.e. carriage return only). Next the list of element names must be modified corresponding to the position numbers which are to be deleted. The operator must press the "LES" function key and enter the symbols of the elements to be removed (lines 6410 to 6440). On return from the subroutine a message is displayed which requires three responses from the operator (line 5540). The list of names in string variable NO\$ is then displayed with an identifying number for each name (lines 5550 to 5610) and the next three requests (lines 5620, 5630 and 5640) are responded to with a number or numbers corresponding to the names desired. Finally two more requests are displayed (lines 5650 and 5660) and the responses to these complete the input required from the operator. Next the names which have been selected are assigned to string variables (lines 5670 to 5710) and printing of the report form begins (lines 5720 to 6010). The sample identification and concentration data for the most recent series of analyses are stored in diskette files by the main program. The filenames differ only in the two number extension (e.g. "SAMPLE.00" to "SAMPLE.09"). The routine next forms the filename and the data for one sample are read into string variable I1\$ and the C array (lines 6020 to 6090). The laboratory identification number and the data are then printed (lines 6095 to 6170). The program loop at lines 6110 to 6130 is a check to determine if any data were selected to be left out. The loop to read data from diskette and print it continues until data for all samples (the number of samples is contained in variable N3) have been obtained. The remainder of the analysis report form is next printed (lines 6190 to 6270) and the routine returns to the main program entry point 3.

ASSEMBLY LANGUAGE ROUTINES

The assembly language routines provide the means for the high level language BASIC to control the operation of the various non-standard input/output devices (see Table 3) which are connected to the system. The routines to operate the standard devices such as the printer, the diskette drives and the keyboard are supplied by the computer manufacturer and are incorporated into the operating system by the user. The set of routines described in this report are incorporated in the BASIC system program and they can be accessed from a BASIC program by CALL statements. The CALL statements provide the number of the routine to be used and any parameters required to pass information to or from the routine (see line 4350 of Appendix A for an example of a CALL statement).

The description of the individual routines which follow are meant to assist a programmer to understand what they accomplish and how to use them. No attempt is made to provide detailed explanations of the code because this would require the reader to have knowledge of the assembly language instruction set for the Nova 2 computer, the protocol required to incorporate the routines into the BASIC system, the interrupt handling procedures and the operating system routines available. Detailed descriptions of these programming aids are given in the various manuals provided by the computer manufacturer^{2,3,4,5}. The line and page numbers given in the following descriptions refer to the commented assembler listing of the routines provided in Appendix C.

GENERAL FORMAT OF ROUTINES

All the routines are assembled together as if they were a single routine, under the mandatory name SBRTB. This name is in fact the symbolic address of the first instruction (line 06 page 0001). Lines 06 to 55 form a table which lists information about each of the routines and the -1 at line 56 signals the end of the table. The information in this table is used by the BASIC interpreter when a CALL statement is encountered. The four pieces of information provided for each routine are 1) the CALL number, 2) the address of the routine, 3) the number of parameters and 4) the type of each parameter. A maximum of eight parameters is allowed and the fourth table entry for each routine is divided into eight two-bit fields. The value of each two-bit field determines the allowable type for

its associated parameter. All parameters used by the present set of routines are of type three which indicates they must be numeric variables. When a CALL statement is encountered in a BASIC program the system places a memory address in accumulator two. This memory address is the first address of a list of addresses of the parameters and the final entry in the list is the return to the BASIC interpreter. The assembly language routines use this mechanism to pass data to and from the BASIC program. The executable code produced by the Assembler program is linked to the BASIC system with the Relocatable Loader program. The routines are described in the order in which they appear in the table on page 0001 of the assembly listing.

ROUTINE 1

Routine 1 at symbolic address ADCB4 (line 57 page 0001) provides a means to read a single selected channel via the analog to digital (A/D) converter and to store the reading in a variable in the BASIC program. The routine requires three parameters which are in order the channel to be read, the gain to be used by the converter and the storage address for the reading. The non-interrupt mode of reading is used, thus once the routine is entered it retains control of the computer until the A/D converter has completed the conversion process (lines 05 and 06 page 0002). All numeric data in a BASIC program is stored in floating-point format and the assembly language routines require it to be in fixed point format. In this routine the conversion to fixed point is accomplished by a subroutine labelled GFIX (line 54 page 0002) and the A/D reading is changed to floating-point format and stored via subroutine STOR (line 57 page 0003).

ROUTINE 10

SETUP (line 29 page 0004) is the symbolic name of this routine which creates a table of channel numbers and associated gain settings required for reading a selected group of channels. The routine requires two parameters, the number of channels to be read and the address of a vector which contains the selected channel numbers and gains. The table created by this routine must be changed every time a new selection of elements is chosen by the operator (see subroutine beginning at statement 2760 in Appendix A). The table is used by another routine which performs multiple

reads of the A/D converter and the entries must conform to the format required by this other routine.

ROUTINE 12

The symbolic address of this routine is AIRDR (line 37 page 0005) and it readies the interrupt system to service interrupts from the A/D converter. It does not pass any data to or from the BASIC program and therefore does not require any parameters. The routine first checks to determine if the A/D converter is already active (lines 45 and 46 page 0005) and if it is not active the interrupt routine is initialized and the A/D converter is started (lines 48 to 60 page 0005) using the first entry from the table created by routine 10. The interrupt handling system services each interrupt until all selected channels have been read (lines 02 to 15 page 0006).

ROUTINE 14

This routine starts at ADCWD (line 28 page 0003) and it converts the A/D readings collected by routine 12 to floating-point format, passes the converted numbers to a vector in the BASIC program and inhibits further interrupts from the A/D converter. The routine requires two parameters, a variable to be used as a flag and the beginning address of a vector in which to store the data. The routine must be used in conjunction with routine 12 (see statements 1060 and 1070 in Appendix A). The routine first checks to see if the A/D converter has completed its task and if it has it is removed from the interrupt system (lines 39 to 42 on page 0003) and the data is changed to floating-point format and passed to a BASIC variable (lines 43 to 56 on page 0003). If the converter was busy the first parameter is set to a non-zero value and the BASIC program can check this flag variable to determine its state.

ROUTINE 20

This routine starts at symbolic address DIOB1 (line 32 page 0002) and provides a means of passing digital data to the computer via the Digital Input/Output device (DIO device). The routine requires two parameters, the number of data items to be transferred and the first storage address. The routine uses the non-interrupt mode of data transfer and the input data

must be in Binary Coded Decimal (BCD) format. The routine is used to transfer data from the Digital Voltmeter (DVM) which was connected to the original mechanical scanning system of the spectrometer. A subroutine (BCDTB line 8 page 0004) converts the BCD input number to binary format and it is then changed to floating-point format (lines 17 to 30 page 0002) and stored in a BASIC variable.

ROUTINE 24

The routine starts a address DIOE2 (line 20 page 0003) and moves digital data out of the computer via the DIO board. The routine requires one parameter, the number to be transferred. The non-interrupt mode of data transfer is used and the output is used to provide a means of opening and closing switches and relays. Each bit in the 16-bit output register can operate a switch or relay, thus the number which is transferred must represent the bits which are to be set or cleared. The number passed from the BASIC program is first converted to fixed format (line 21 page 0003) and is then transferred to the DIO board (line 22 page 0003).

ROUTINE 3

The routine which starts at address IOPIT (line 27 page 0008) attaches two devices, the Programmable Interval Timer (PIT) and the I/O Subsystem Controller to the interrupt system. The routine does not require any parameters and provides some initialization code for the controller. The interrupt handling routines to operate the two devices are included in this section of the code (lines 23 to 58 page 0006, 6 to 60 page 0007 and 1 to 20 page 0008).

ROUTINE 30

This routine begins at address SETEM (line 56 page 0008) and creates a table with pairs of data which provide time intervals and switch settings. The routine requires one parameter, the beginning address of a vector in the BASIC program which contains the selected pairs of numbers. The table created by this routine is used to operate the interval timer and the triac output module of the controller to provide timed sequential operation of the spectrometer system. The sequence of operations is started by the push of the start button on the spectrometer which causes an interrupt and

starts the interval timer. At the end of each time interval an interrupt from the interval timer causes the beginning of the next time interval and associated triac switch settings. In the current version of the routine a maximum of eight time intervals is allowed and the maximum time for each interval is 65 seconds. The BASIC vector which supplies the data must contain the time intervals in the first eight locations and the numbers representing the switch settings in the following eight locations. The time intervals must be entered as negative numbers.

ROUTINE 33

The final routine begins at address TRISET (line 26 page 0009) and provides a means to set the triac switches independently. The routine requires one parameter, a number which represents the switches to be set or cleared. The number is first converted to fixed format (line 34 page 0009) and then passed to the triac module (line 36 page 0009).

DISCUSSION

The operating program has been used by three different operators for extended periods of time during the past three years. All of the operators have found the program easy to use and none have seen the necessity for any major changes. The only significant change in the program during this period was the addition of the function "FAC" which applies correction factors to the data based on measurements made on a secondary standard.

The routine to provide a formatted analysis report is not currently being used because a similar report is produced from data entered into the Departmental mainframe computer. The routine is not entered with the main program but is stored on diskette as a separate file (filename is "REPORTWR.LS") which can be merged with the main program if desired. Also the portion of the main program which stores the analyses data on diskette for use by the report writing routine is currently bypassed. The program statements were retained and their action can be restored by removing two program lines. The report writing routine is the one part of the program which could be considerably improved by making it more general and flexible. If such a change is made in the future it is quite likely that

the routine would be developed as a separate program rather than as a function available from the main program.

The 64 kilobytes of memory are sufficient to operate the program without the necessity to resort to the use of overlays or program chaining. When the entire program is loaded there are 1380 bytes of memory left for program or data use and when the report writing routine is deleted there are 5036 bytes available.

Addition of a graphics capability would enhance the system but this would require major hardware and software changes which do not seem justified. Since the program has been operating satisfactorily over the past three years and appears to provide a satisfactory complement of functions there are no plans at the present to make any major changes.

REFERENCES

1. Knuth, D.E. The Art of Computer Programming
Volume 1 Fundamental Algorithms
Addison-Wesley Publishing Company
2. RDOS/DOS Macroassembler User's Manual
093-000081-05 Data General Corporation
3. Extended BASIC User's Manual
093-000065-09 Data General Corporation
4. User's Manual, Programmer's Reference Peripherals
014-000632-00 Data General Corporation
5. Disk Operating System (DOS) Reference Manual
093-000201-03 Data General Corporation

TABLE 1. ASSEMBLY LANGUAGE ROUTINES

CALL #	FUNCTION	PARAMETERS
1	Read ADC, 1 channel, non-interrupt	(3) Channel #, gain, data address
10	Merge channel #'s and gains for multiple ADC reads, interrupt	(2) No. of elements, address of gain & Channel #'s
12	Identify ADC for interrupts	none
14	Convert ADC data to floating point and store in data array	(2) Busy flag, data address
20	BCD input to DIO non-interrupt	(2) No. data items, data address
24	Set bits in DIO output register	(1) No. representing bits to set
3	Identify programmable timer and I/O subsystem for interrupts	none
30	Set time intervals and switch settings in table for sequencing	(1) Address of data to be used
33	Set triac switches	(1) No. representing bits to set

NOTE:

ADC = Analog to Digital Converter

BCD = Binary Coded Decimal

DIO = Digital Input Output

TABLE 2. FUNCTION KEY ROUTINES

MNEMONIC	CHARACTER DISPLAYED	FUNCTION	STATEMENT NUMBER
AVG	q	Average of Data	1480
ERA	r	Erase Data and Print Heading	640
RSD	s	% Relative Standard Deviation	1480
DEL	t	Delete Lines of Data	1680
MOR	u	Add Elements to List	560(2480)
LES	v	Remove Elements From List	1780
PRT	w	Print Results	3470
RES	x	Restore Data to Display	1360
FAC	y	Calculate and Print Correction Factors	5210
STD		Standardization	4460
(CTRL)RES	8	Print Data (Analagous to RES)	5000
PAR	z	Print Analysis Report	5340

TABLE 3. NON-STANDARD I/O DEVICES

DEVICE	FUNCTION
Analog to Digital Converter and Multiplexor	Converts voltage signals (0-10 V, DC) to digital numbers under program control. 32 channels can be read sequentially or randomly.
Digital Input/Output Board	Provides 16 TTL compatible input lines and 16 output lines for transfer of digital data.
Programmable Interval Timer	Provides program selectable time intervals from one millisecond to 65 seconds. An interrupt occurs at the end of the interval.
I/O Subsystem Controller	Provides program control and power for up to 16 I/O modules which plug into slots in the controller crate.
General Purpose Input Module (uses I/O controller)	Receives digital data on 16 independent lines. The inputs may be from 6 to 55 V DC or 15 to 135 V AC.
Triac Output Module (uses I/O controller)	Controls the state of 16 triac switches for switching 25-135 V AC.
Digital to Analog Converter (uses I/O controller)	Provides analog voltage output on four program selectable output lines. Three output voltage ranges are available all with 12-bit resolution.

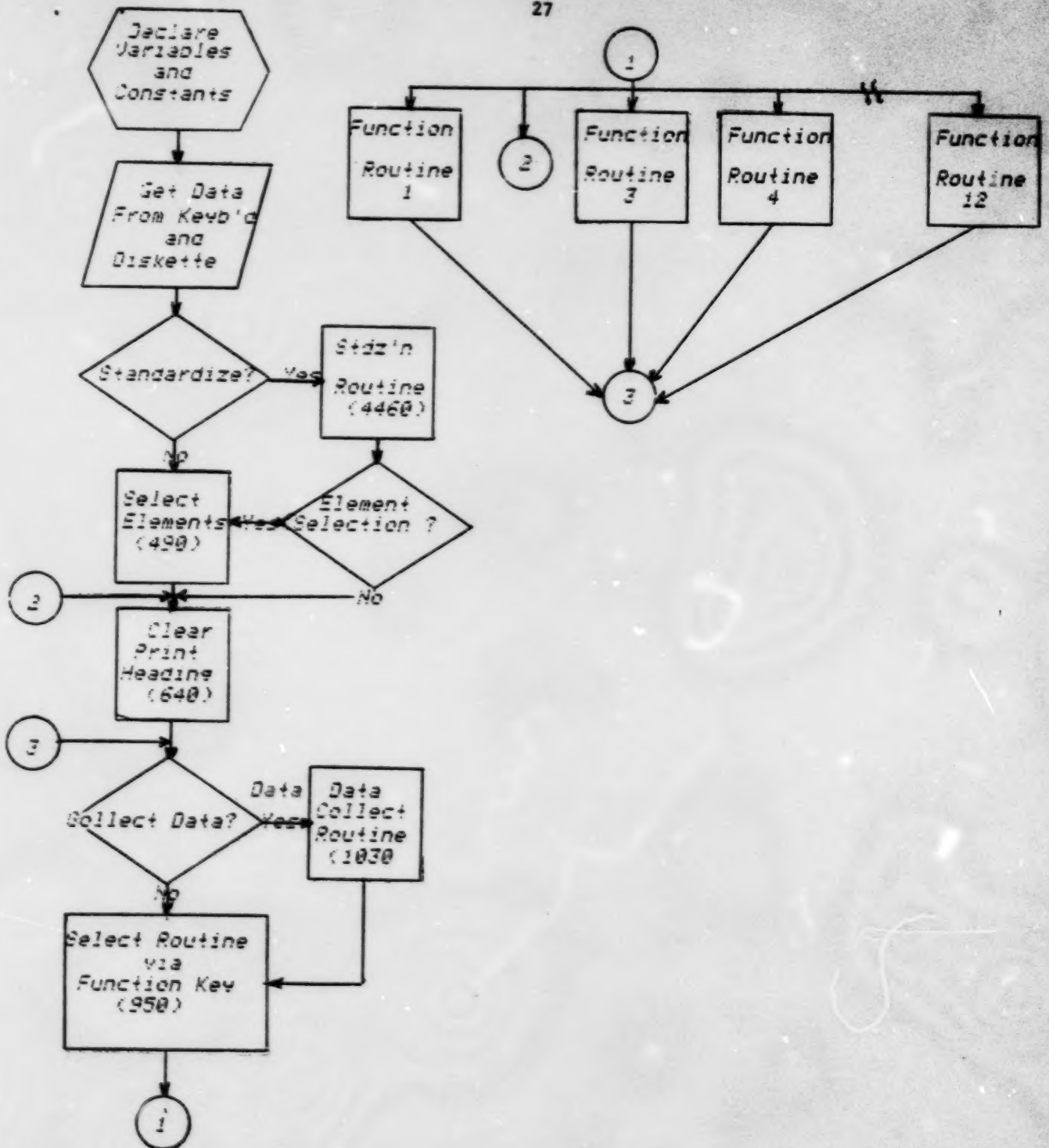


FIG. 1. MAIN PROGRAM FLOWCHART

1	2	3	4	5	6	7	8	9	10	11	12	13	14
Link	a0	a1	a2	a3	Cur.	Cal.	Rdg.	Conc.	Chnl.	Rdg.	Conc.	Gain	Coeff

L.S. Fit Coefficients Secondary Std. Low Limit High Limit

FIG. 2. COLUMN IDENTIFICATION FOR S ARRAY

1	2	3	-	-	N	N+1	-	-	-	2N	2N+1	-	-	3N	-	-	140
---	---	---	---	---	---	-----	---	---	---	----	------	---	---	----	---	---	-----

Cumulative Totals First Data Area Second Data Area Mth Data Area

N = No. of Data Items
 M = No. of Replicates of Data Items

FIG. 3. DATA ARRANGEMENT IN D ARRAY

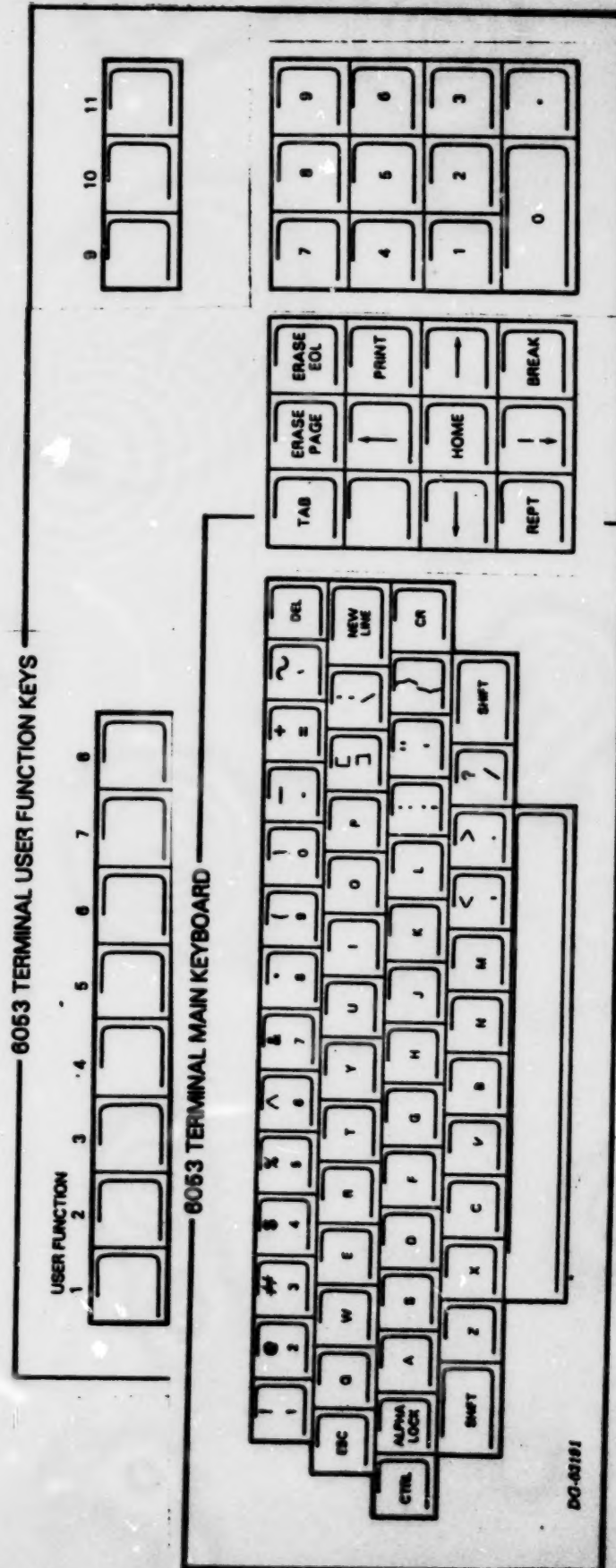


FIG. 4. TERMINAL KEYBOARD LAYOUT

MINERAL SCIENCES LABORATORIES LABORATOIRE DES SCIENCES MINÉRALES

Report No.
No. de Rapport: 84-1237

Date: 23 / 7 / 1984

Samples/
Echantillons: LOW ALLOY STEEL

SPECTROCHEMICAL ANALYSIS/
ANALYSE SPECTROCHIMIQUE: -Quantometer

From: A. R. PALMER

Project No.
No. de Projet: 378119

Element - %

30

Lab No.	Product: Produit	C	S	P	SI	MN	AL	SN	CU	NI	CR	MO	U
16029	INGOT SLICE 1572 T	0.710	0.018	0.010	0.186	0.788	0.015	0.014	0.033	0.044	0.047	0.010	0.005
16030	INGOT SLICE 1572 TE	0.434	0.015	0.010	0.183	0.755	0.023	0.012	0.031	0.040	0.047	0.010	0.005
16031	INGOT SLICE 1572 MC	0.326	0.016	0.010	0.186	0.763	0.017	0.013	0.032	0.040	0.050	0.010	0.005
16032	INGOT SLICE 1573 TC	0.328	0.014	0.011	0.197	0.760	0.018	0.015	0.031	0.040	0.044	0.010	0.005
16033	INGOT SLICE 1573 TE	0.325	0.014	0.010	0.197	0.749	0.019	0.017	0.030	0.040	0.044	0.010	0.005
16034	INGOT SLICE 1573 MC	0.335	0.014	0.010	0.203	0.777	0.019	0.016	0.032	0.040	0.045	0.010	0.005

Analyst/Analyste: T. R. CHURCHILL

Approved/Approuvé: Section Head: Chef de Section

Copies to/s: 1. R. K. BUHR
2. P. J. LACROIX
3. R. G. SABOURIN

4. Section Files: Dossier de Section
5. Laboratory Files: Dossiers de Laboratoire

FIG. 5. TYPICAL ANALYSIS REPORT

APPENDIX A. MAIN PROGRAM LISTING

```

0010 REM ***** "OP4REM.CI" *****
0012 REM COMMENTED VERSION OF OP4.CI
0013 REM AS OF JUNE 1984
0015 REM OP4.CI IS THE CURRENT PROGRAM TO OPERATE THE APPLIED RESEARCH
0020 REM LABORATORIES MODEL 29000/29710 QUANTOMETER (UVAQ)
0030 REM THE PROGRAM PROVIDES FOR TREATMENT OF THE COLLECTED DATA
0040 REM AND HAS TWELVE ROUTINES WHICH CAN BE ACCESSED VIA FUNCTION KEYS
0050 REM
0085 REM =====
0086 REM *****
0087 REM =====
0090 REM
0095 REM ***** DIMENSION ARRAYS & STRINGS & ASSIGN CONSTANTS *****
0100 REM
0110 DIM A(63),B(30),C(25),D(140),E(14),F(10)
0120 DIM G(25),R(25),S(30,14),X(65),Z(65)
0130 DIM A2$(4),B0$(20),C1$(20),D1$(21),F0$(5),F1$(13),I1$(40)
0140 DIM L0$(180),L2$(170),M0$(130),M1$(36),N6$(10),P0$(4),R0$(4),S0$(10)
0150 LET B0$="abcdefghijklmnopqrstuvwxyz(<18>8"
0160 LET D1$="SUNMONTUENEDTHUFRISAT"
0170 LET F4,F5,F7=0
0180 LET F0$=" "
0190 LET M0$(1,42)=" COCO1 C C1 S P SISI1MN2 MNMN1 ALAL1AL2"
0200 LET M0$(43,84)=" SNSN1 CUCU1 NINI1 CRCR1 MO U TITI1 NB ZR"
0210 LET M0$(85,126)=" B MG NMG2 PBPB1 ZNZN1ZN2 AGAG1 FEFE1 CA"
0220 LET M1$="JANFEBMARAPR MAYJUNJUL AUGSEP OCTNOV DEC"
0230 LET N,N3=1
0240 LET N6$="0123456789"
0250 LET P=99
0260 LET P0$="(<12>)"
0270 LET R0$="(<13>)"
0280 LET S0$="SAMPLE # "
0290 MAT B=CON
0295 REM =====
0300 REM
0305 REM ***** ROUTINE TO READ DATA FROM DISKETTE TO S ARRAY & B VECTOR *****
0307 REM
0310 PRINT "LAS(1), C.I.(2), HAS(3), HI CR HAS(4), TS(5), CU/NI(6)"
0320 INPUT A1
0322 LET A1=ABS(A1)
0324 IF A1>4 THEN GOTO 0310
0325 LET C1$="D4D5D2D1"
0330 LET F1$="ALLOYTYPE.",C1$(A1*2-1,A1*2)
0340 OPEN FILE (0,3),F1$,56
0350 FOR J0=0 TO 30
0352 MAT READ FILE (0,J0),E
0355 FOR J1=1 TO 14
0370 IF EOF(0) THEN GOTO 0450
0390 LET S(E(1),J1)=E(J1)
0400 NEXT J1
0410 NEXT J0
0450 CLOSE FILE (0)
0452 INPUT "RECOVER B COEFF'S NO(0),YES(<>0) ?",M1:
0455 IF M1 THEN GOSUB 4870 !RECOVER B COEFF'S
0456 REM =====
0457 REM
0458 REM ***** ROUTINE FOR ELEMENT SELECTION *****
0459 REM
0460 IF (3-A1)<=0 THEN LET L2$=" C, S, P, SI, MN, NI, CR, MO" ! HI ALLOY
0470 IF A1=2 THEN LET L2$=" C, S, P, SI, MN, NI, MG" ! CAST IRON

```

```

0480 IF A1<>1 THEN GOTO 0540 ! SELECT DESIRED ELEMENTS
0485 PRINT "CALIBRATE(0), STD. ELEMENTS(1), STD. +OTHERS(2), YOUR CHOICE(3) ?"
0490 INPUT A2
0500 IF ABS(A2)>3 THEN GOTO 0490
0510 IF ABS(A2)=0 THEN GOTO 4460 ! STANDARDIZATION
0520 IF ABS(A2)=3 THEN GOTO 0560 ! SELECT ADDITIONS
0530 LET L2$=" C, S, P, SI, MN, AL, SN, CU, NI, CR, MO, U" ! STD. ELEMENTS
0540 GOSUB 2480 ! * * * TABLE LINKAGE * * *
0550 IF ABS(A2)<>2 THEN GOTO 0640
0552 REM =====
0554 REM
0555 REM ***** GET SYMBOLS TO ADD TO CURRENT ELEMENT LIST *****
0557 REM
0560 INPUT "ELEMENTS ARE ? ",L2$
0570 GOSUB 2480 ! ADD ELEMENTS TO LIST
0620 INPUT "CHANGES? ",A2$
0630 IF LEN(A2$) THEN GOTO 0940
0632 REM =====
0634 REM
0635 REM ***** ROUTINE TO ERASE DATA & PRINT ELEMENT HEADING *****
0637 REM USER FUNCTION KEY 2 (ERA) SYMBOL "r"
0640 MAT D=ZER ! CLEAR DATA AREA & FLAGS
0680 LET F8,F9=0
0700 PRINT "<12><28>" ! ERASE PAGE, DIM ON
0730 MAT F=ZER
0760 MAT X=ZER
0770 MAT Z=ZER
0800 GOSUB 2355 ! * * * DATE CALC.
0820 PRINT USING "### ##/### /####",D1*(N2*3+1,N2*3+3),SYS(1),
N1*(C4*3+1,C4*3+3),SYS(3)
0830 PRINT USING "##:##:##",SYS(11),SYS(12),SYS(13) ! HRS., MINS., SECS.
0840 PRINT "<29>" ! DIM OFF
0860 GOSUB 4800 ! PRINT ELEMENTS
0880 IF F7<>2 THEN GOTO 0910 ! F7=2, STANDARDIZATION
0890 PRINT "!!!<14><20>SPARK CHK. SAMPLE 3 OR MORE TIMES<21><15><3>"
0900 GOTO 0930
0910 INPUT "----> <14> <20>SAMPLE NUMBER<21>:<15><3>",I1$
0920 PRINT "<4>"; ! INHIBIT BLINK
0925 REM =====
0927 REM
0930 INPUT A2$ ! RETURN POINT FOR ALL ROUTINES
0935 IF LEN(A2$)=0 THEN GOTO 1030
0940 IF A2$(1,1)<>"<30>" THEN GOTO 1030 ! IS THIS A USER FUNCTION KEY
0942 REM =====
0944 REM
0945 REM ***** ROUTINE TO CHECK FOR USER FUNCTION KEYS *****
0947 REM
0950 LET C1=POS(B0$,A2$(2,2),0) ! FIND FUNCTION KEY POS'N
0960 IF C1=0 THEN GOTO 1010
0970 ON C1 THEN GOTO 1480, 0640, 1480, 1680, 0560, 1780, 3470, 1360,
5200, 1010, 1010, 4460, 5000, 5300
1010 PRINT "ROUTINE NOT AVAILABLE"
1020 GOTO 0930
1022 REM =====
1023 REM
1025 REM ***** ROUTINE TO COLLECT DATA FROM SPECTROMETER *****
1027 REM
1030 FOR M=1 TO 10
1032 IF F(M)=0 THEN GOTO 1040 ! F(M)=0 =DATA AREA AVAILABLE
1034 NEXT M

```

```

1036 PRINT "<20>***<14>BUFFER FULL<15>***<21><3>"
1039 GOTO 0930
1040 LET L=M*N+1 ! INDEX TO DATA AREA
1042 IF (L+N)>140 THEN GOTO 1036 ! DATA MAX.=140
1045 IF L/(N+1)>10 THEN GOTO 1036
1050 GOSUB 4320 ! * * * READ INT. STD. CHNL.
1055 FOR L0=1 TO 20 ! 20 READS
1060 CALL 12 ! ENABLE A TO D INTERRUPTS
1065 LET F1=0
1070 CALL 14,F1,A(1) ! READ A TO D
1075 IF F1 THEN GOTO 1070 ! IF F1 SET, READ IS BUSY
1080 FOR I=1 TO N-1
1085 LET D(I+L)=D(I+L)+A(I) ! READ N CHNL'S
1090 NEXT I
1095 NEXT L0
1097 LET T=P ! SET POINTER TO LINKED LIST
1100 FOR I=1 TO N-1
1105 LET D(I+L)=D(I+L)/20 ! AVG. OF 20 READS
1106 IF D(I+L)<4094 THEN GOTO 1110
1107 LET D(I+L)=1000 ! OVERFLOW
1108 GOTO 1120
1110 LET D(I+L)=D(I+L)*B(I+1)*S(T,7)/(4.1*S(T,6)) ! APPLY CORRECTIONS
1120 LET D(I+1)=D(I+1)+D(I+L) ! CUMULATION OF BURNS
1130 LET X(I+1)=X(I+1)+D(I+L)^2
1135 IF (I-INT(I/10)*10)=0 THEN PRINT USING " ###.##",D(I+L)
1137 IF (I-INT(I/10)*10)=0 THEN GOTO 1150
1140 PRINT USING " ###.## ",D(I+L)
1150 LET T=SCT,1 ! SET POINTER FOR NEXT ITEM
1155 NEXT I
1160 LET N0=1
1162 CALL 24,N0 ! CLOSE RESET RELAY
1164 DELAY =1
1166 LET N0=0
1170 CALL 24,N0 ! OPEN RESET
1180 LET J1=INT(L/N)
1190 LET F(J1)=J1 ! SET DATA FLAG
1330 GOTO 0930
1332 REM =====
1335 REM
1340 REM ***** RESTORE ROUTINE PRINTS HEADING & EACH BURN ON VIDEO *****
1350 REM USER FUNCTION KEY 8 (RES) SYMBOL "x"
1355 REM
1360 GOSUB 4800 ! PRINT ELEMENT HEADING
1380 FOR M=1 TO 10
1390 IF F(M) THEN GOTO 1420 ! IF F(M) SET THERE IS DATA
1400 NEXT M
1410 GOTO 0930
1420 LET L0=M*N+1 ! INDEX FOR D VECTOR
1425 PRINT USING "(#)",M
1430 FOR I=L0+1 TO L0+N-1
1435 IF ((I-L0)-INT((I-L0)/10)*10)=0 THEN PRINT USING " ###.##",D(I)
1437 IF ((I-L0)-INT((I-L0)/10)*10)=0 THEN GOTO 1450
1440 PRINT USING " ###.## ",D(I)
1450 NEXT I
1460 PRINT
1470 GOTO 1400
1475 REM =====
1477 REM
1478 REM ***** ROUTINE TO DETERMINE AVG.(GOSUB CONCn, CALCN) & RSD *****
1479 REM USER FUNCTION KEYS 1 & 3 (AVG & RSD) SYMBOLS "q" & "s"

```

```

1480 LET J2=0
1490 LET F8=1 ! FLAG SET = AVG. CALCULATED
1500 FOR I=1 TO 10
1510 IF F(I) THEN LET J2=J2+1 ! CALC. NO. OF BURNS
1520 NEXT I
1530 IF F7=2 THEN IF J2<3 THEN GOTO 0890 ! 3 BURNS MIN. FOR STANDARDIZ'N
1540 PRINT "AVG.(OF",J2,")"
1550 LET C(I)=D(I)/J2 ! AVG. FOR INT. STD.
1555 LET Z(I)=X(I)/J2 ! AVG. SUM OF SQUARES
1560 FOR I=2 TO N
1570 LET C(I)=D(I)/J2 ! DATA AVG.
1575 IF D(I)/J2=1000 THEN LET C(I)=1000
1580 LET Z(I)=X(I)/J2 ! AVG. SUM OF SQUARES
1582 LET A(I)=C(I)
1585 IF ((I-1)-INT((I-1)/10)*10)=0 THEN PRINT USING " ###.##",C(I)
1587 IF ((I-1)-INT((I-1)/10)*10)=0 THEN GOTO 1610
1590 PRINT USING " ###.## ",C(I)
1610 NEXT I
1620 IF A2*(2,2)="s" THEN GOTO 3360 ! CALC. RSD.
1630 IF F7=2 THEN GOTO 4510 ! GOTO CALIBRATION ROUTINE
1640 GOSUB 2940 ! * * * CALC. CONC'N
1650 IF A1=5 THEN GOTO 2100 ! CORRECT'N FOR TOOL STEEL
1660 IF A1>2 THEN GOTO 2110 ! HI ALLOY,USE RATIO METHOD
1670 GOTO 0930
1672 REM =====
1674 REM
1675 REM ***** ROUTINE TO DELETE LINES of DATA *****
1677 REM USER FUNCTION KEY 4 (DEL) SYMBOL "+"
1678 REM
1680 LET A2*=""
1685 INPUT "LINE # ",A2*
1687 IF LEN(A2*)=0 THEN GOTO 0930
1688 IF A2*(1,1)<>"<30>" THEN GOTO 1692 ! IF NOT A FUNCT'N KEY
1690 PRINT
1691 GOTO 0950 ! IT IS A FUNCT'N KEY
1692 LET N1=VAL(A2*) ! LINE NO.
1693 IF N1>10 THEN GOTO 1680
1700 LET F(N1)=0
1710 LET L=N1*N ! CALC. INDEX
1720 FOR I=1 TO N
1730 LET D(I)=D(I)-D(I+L) ! ADJUST CUMUL. TOTAL
1740 LET X(I)=X(I)-D(I+L)^2 ! ADJUST SUM OF SQUARES
1750 LET D(I+L)=0 ! ZERO DATA AREA
1760 NEXT I
1770 GOTO 1680
1775 REM =====
1776 REM
1778 REM ***** ROUTINE TO REMOVE ELEMENTS FROM CURRENT LIST *****
1779 REM USER FUNCTION KEY 5 (LES) SYMBOL "v"
1780 INPUT "REMOVE ?",L2* ! USE CORRECT FORMAT
1790 FOR I=1 TO LEN(L2*)-2 STEP 4
1800 LET P3=POS(M0*,L2*(I,I+2),0) ! FIND POS'N IN MASTER LIST
1810 IF P3=0 THEN GOTO 2025 ! NOT A VALID SYMBOL
1820 LET J=(P3+2)/3 ! POS'N IN S ARRAY (I.E. LINK)
1830 IF J=P THEN GOTO 1960
1840 LET T=P
1850 IF J=S(T,1) THEN GOTO 1980 ! ADJUST LINK VALUE
1860 LET T=S(T,1)
1870 IF T=99 THEN GOTO 2000
1880 GOTO 1920

```

```

1960 LET P=S(P,1)
1970 GOTO 1990
1980 LET S(T,1)=S(J,1)
1990 LET N=N-1
2000 NEXT I
2010 GOSUB 2770 ! FORM TITLE LINE SET CHNLS. & GAINS
2020 GOTO 0620
2025 PRINT L2$(I,I+2):" Not a Valid Symbol Dum Dum !!!"
2030 GOTO 2000
2035 REM =====
2040 REM
2050 REM ***** ROUTINE FOR CONCN. CALCn. USING RATIO METHOD *****
2060 REM
2100 LET C(5)=C(5)-(565.3*C(8)+910*C(9))/71891
2110 LET D1=C(1)+C(2)+C(3)+C(4)+C(5)+C(6)+C(7)+C(8)+C(9)+C(10)+1 ! CONCN'S SUM
2120 LET F3=99/D1
2130 PRINT F3
2140 FOR I=2 TO N
2150 LET R(I)=C(I)
2160 LET C(I)=C(I)*F3
2162 IF ((I-1)-INT((I-1)/10)*10)=0 THEN PRINT USING " ###.##",C(I)
2165 IF ((I-1)-INT((I-1)/10)*10)=0 THEN GOTO 2180
2170 PRINT USING " ###.### ",C(I)
2180 NEXT I
2190 PRINT
2200 GOTO 0930
2210 REM =====
2220 REM
2250 STOP
2275 REM * * * SUBROUTINE TO PRINT DATE, TIME, ELEMENT NAMES ON PRINTER * * *
2277 REM
2280 OPEN FILE (1,1), "$TT01"
2285 PRINT FILE (1), USING "###.###.###.###", D1$(N2*3+1, N2*3+3),
M1$(C4*3+1, C4*3+3), SYS(1), SYS(3)
2290 PRINT FILE (1), USING "Time: - ###.##", SYS(11), SYS(12) ! HRS. & MINS.
2295 CLOSE FILE (1)
2300 OPEN FILE (1,1), "$TT01"
2305 PRINT FILE (1), TAB(10); L0$(1, LEN(L0$)-INT(LEN(L0$)/112)*(LEN(L0$)-112))
2307 IF LEN(L0$)<=112 THEN GOTO 2327
2310 CLOSE FILE (1)
2320 OPEN FILE (1,1), "$TT01"
2325 PRINT FILE (1), TAB(10); L0$(113, LEN(L0$))
2327 PRINT FILE (1), R0$
2330 CLOSE FILE (1)
2335 RETURN
2337 REM =====
2340 REM
2345 REM ***** SUBROUTINE TO CALC. DAY & MONTH CONSTANTS *****
2350 REM
2355 LET C4=SYS(2)-1
2360 IF SYS(2)>2 THEN GOTO 2400
2370 LET G1=SYS(3)-1
2380 LET F1=SYS(2)+13
2390 GOTO 2420
2400 LET G1=SYS(3)
2410 LET F1=SYS(2)+1
2420 LET N1=INT(365.25*G1)+INT(30.6*F1)+SYS(1)-621049
2430 LET N2=(N1/7-INT(N1/7))*7+.01
2440 RETURN
2450 REM =====

```

```

2455 REM
2460 REM ***** ROUTINE TO ADD SELECTED ELEMENTS TO CURRENT LIST *****
2470 REM USER FUNCTION KEY 5 (MOD) SYMBOL "u"
2475 REM
2480 LET N4=0
2485 FOR I=1 TO LEN(L2$)-2 STEP 4
2490 LET P3=POS(M0$,L2$(I,I+2),N4) ! FIND ORDINAL POS'N IN M0$
2500 IF P3=0 THEN GOTO 2750
2510 LET J=(P3+2)/3 ! POS'N IN S ARRAY
2520 IF J=P THEN GOTO 2730 ! ADJUST LINKS
2600 IF J<P THEN GOTO 2700
2610 LET J1=P
2620 LET T=S(J1,1)
2630 IF J<T THEN GOTO 2670
2640 IF J=T THEN GOTO 2730
2650 LET J1=T
2660 GOTO 2620
2670 LET S(J1,1)=J
2680 LET S(J,1)=T
2690 GOTO 2720
2700 LET S(J,1)=P
2710 LET P=J
2720 LET N=N+1
2725 IF F7=2 THEN LET N4=J*3 ! ELEMENTS IN ORDER, ADVANCE SEARCH POS'N
2730 LET S(J,5)=S(J,7) ! SET FACTOR TO 1
2735 NEXT I
2740 GOSUB 2770 ! FORM TITLE LINE & SET GAINS & CHNL'S
2745 RETURN
2750 PRINT "You Goofed ";L2$(I,I+2); " Not a Valid Symbol"
2752 GOTO 2735
2755 REM =====
2757 REM
2760 REM * * * * * SUBROUTINE TO FORM TITLE LINE & SET CHNL. #'S & GAINS * * *
2765 REM
2770 LET P1=P
2780 LET F4,F6,L1=0
2795 LET L0$=""
2800 LET L0$(8*L1+1,8*L1+8)=F0$(1,2),M0$(3*P1-2,3*P1),F0$(3,5) ! FORM
ELEMENT SYMBOL
2810 LET A(L1+1)=S(P1,13) ! PUT GAIN IN A ARRAY
2820 LET A(L1+33)=S(P1,10) ! PUT CHNL. NO. IN A ARRAY
2830 LET L1=L1+1
2835 LET B(L1+1)=S(P1,14) ! SET B COEFF.
2840 IF P1=7 THEN LET F4=L1+1 ! F4= POS'N OF SI IN LINKED LIST
2850 IF P1=23 THEN IF F4 THEN LET F6=L1+1 ! F6 = POS'N OF M0 IN LINKED LIST
2860 LET P1=S(P1,1)
2870 IF P1<>99 THEN GOTO 2800
2880 LET A(L1+1)=P1 ! 99 INDICATES END OF LIST
2885 LET N0=N-1 ! NO. OF ELEMENTS LESS FE
2890 CALL 10,N0,A(1) ! SET CHNL'S & GAINS
2900 RETURN
2905 REM =====
2910 REM
2920 REM * * * * * SUBROUTINE TO CALCULATE CONCENTRATIONS * * * *
2930 REM
2940 LET T=P
2960 PRINT
2970 PRINT "CONC."
2980 FOR I=2 TO N
2990 IF C(I)<S(T,8) THEN GOTO 3090 ! IF VALUE LESS THAN MIN.

```

```

3000 IF C(I)>S(T,11) THEN GOTO 3120 ! IF VALUE > MAX.
3010 LET X1=(C(I)-S(T,2))/S(T,3) ! 1ST ESTIMATE (1ST ORDER FIT)
3020 LET Y1=S(T,2)+S(T,3)*X1+S(T,4)*X1^2 ! CALC. Y FOR ESTIMATED X
3030 IF Y1<=0 THEN GOTO 3070
3040 IF ABS(1-Y1/C(I))<.002 THEN GOTO 3150 ! IF <.2% DIFF., FINISHED
3050 LET X1=X1*(C(I)-S(T,2))/(Y1-S(T,2)) ! NEXT ESTIMATE
3060 GOTO 3020
3070 PRINT " ERROR "
3080 GOTO 3180
3090 IF ((I-1)-INT((I-1)/10)*10)=0 THEN PRINT USING "<#.####",S(T,9) !END
OF LINE
3095 IF ((I-1)-INT((I-1)/10)*10)=0 THEN GOTO 3100
3097 PRINT USING "<#.#### ",S(T,9) ! NOT LINE END
3100 LET C(I)=S(T,9) ! SET CONC. =MIN.
3110 GOTO 3180
3120 IF ((I-1)-INT((I-1)/10)*10)=0 THEN PRINT USING ">#.### ",S(T,12)
3125 IF ((I-1)-INT((I-1)/10)*10)=0 THEN GOTO 3130
3127 PRINT USING ">#.### ",S(T,12)
3130 LET C(I)=S(T,12) ! SET CONC. = MAX.
3140 GOTO 3180
3150 IF X1<S(T,9) THEN GOTO 3090 ! IF CONC. < MIN.
3160 IF ((I-1)-INT((I-1)/10)*10)=0 THEN PRINT USING " ##.###",X1
3165 IF ((I-1)-INT((I-1)/10)*10)=0 THEN GOTO 3170
3167 PRINT USING " ##.### ",X1
3170 LET C(I)=X1 ! PUT CONC. IN C(I)
3180 LET T=S(T,1) ! NEXT ELEMENT
3190 NEXT I
3200 PRINT
3210 IF F6=0 THEN RETURN ! NO CORRECTION FOR HQ ON SI
3250 LET Z0=S(7,2)+S(7,5)*C(F6)-A(F4) ! SI CORRECTION
3260 LET C(F4)=-Z0/S(7,3)
3270 FOR I=2 TO N ! DISPLAY CONC. VALUES
3272 IF ((I-1)-INT((I-1)/10)*10)=0 THEN PRINT USING " ##.###",C(I)
3275 IF ((I-1)-INT((I-1)/10)*10)=0 THEN GOTO 3290
3280 PRINT USING " ##.### ",C(I)
3290 NEXT I
3300 PRINT
3320 RETURN
3325 REM =====
3330 REM
3340 REM * * * * CALCULATION OF RELATIVE STANDARD DEVIATION * * *
3350 REM
3360 PRINT "<13>R.S.D."
3370 LET F9=1
3380 FOR I=2 TO N
3390 LET G(I)=SQR(ABS(Z(I)-C(I)^2))*SQR(J2/(J2-1))*100/C(I)
3400 IF G(I)<.1 THEN GOTO 3430
3410 PRINT USING " ##.## ",G(I)
3420 GOTO 3440
3430 PRINT USING " #.###% ",G(I)
3440 NEXT I
3450 GOTO 1630
3455 REM =====
3457 REM
3460 REM ***** ROUTINE TO PRINT AUG'S & CONCns. ON PRINTER *****
3465 REM USER FUNCTION KEY 7 (PRT) SYMBOL "U"
3467 REM
3470 IF F8>0 THEN GOTO 3500 ! F8 SET=AUG. CALC.
3480 PRINT "* * *- 'AUG.' REQUIRED BEFORE 'PRT' * * *"
3490 GOTO 0930

```

```

3500 PRINT "REPLY: - '1' FOR HEADER + AVERAGES + CONCENTRATIONS"
3510 PRINT "          '2' FOR ABOVE WITH R.S.D.'S"
3520 PRINT "          '3' FOR AVERAGES + CONCENTRATIONS ONLY"
3530 PRINT "          '4' AS ABOVE WITH R.S.D.'S"
3535 INPUT "RESPONSE : ", F0
3540 IF F0/2<>INT(F0/2) THEN GOTO 3580 ! IF F0 EVEN CHECK FOR RSD.
3550 IF F9 THEN GOTO 3580 ! F9 SET= RSD CALC'D
3560 PRINT " * * * 'RSD' REQUIRED BEFORE 'PRT' * * *"
3570 GOTO 0930
3580 ON F0 THEN GOTO 3590, 3590, 3700, 3700
3590 PRINT "SET PRINTER TO TOP OF PAGE IF DESIRED & PRESS 'CR'";
3595 INPUT " ON VIDEO TERMINAL ", A2$
3600 GOSUB 2355 ! * * * DATE CALC'N
3610 GOSUB 2280 ! * * * PRINT HEADING ! ON PRINTER
3700 OPEN FILE (1,1), "$TT01"
3710 PRINT FILE (1), I1$, R0$ ! SAMPLE IDENT'N, CR
3720 IF A1>2 THEN PRINT FILE (1), USING " %F0=##.## ", F3 ! A1>2=RATIO METHOD
3730 IF F0/2<>INT(F0/2) THEN GOTO 3830 ! IF ODD, AVG. ONLY
3750 PRINT FILE (1), " RSD. "
3760 FOR J=2 TO N ! PRINT RSD. VALUES
3780 PRINT FILE (1), USING " ##.##% ", G(J);
3785 IF J<>15 THEN GOTO 3810
3787 IF N=15 THEN GOTO 3810
3790 PRINT FILE (1), R0$; TAB(11);
3810 NEXT J
3820 PRINT FILE (1), R0$;
3830 CLOSE FILE (1)
3840 OPEN FILE (1,1), "$TT01"
3850 PRINT FILE (1), R0$;
3860 PRINT FILE (1), USING "AVG.(#) ", J2; ! J2=NO. OF BURNS AVG'D
3880 FOR L0=2 TO N ! PRINT AVG.'S
3890 PRINT FILE (1), USING " ###.## ", A(L0);
3892 IF L0<>15 THEN GOTO 3900
3893 IF N=15 THEN GOTO 3900
3894 PRINT FILE (1), R0$; TAB(11);
3900 NEXT L0
3910 PRINT FILE (1), R0$;
3920 CLOSE FILE (1)
3930 IF A1<3 THEN GOTO 4030
3940 OPEN FILE (1,1), "$TT01"
3950 PRINT FILE (1), R0$;
3960 PRINT FILE (1), " RATIO";
3980 FOR L0=2 TO N ! PRINT RATIOS
3990 PRINT FILE (1), USING " .#### ", R(L0);
3992 IF L0<>15 THEN GOTO 4000
3993 IF N=15 THEN GOTO 4000
3994 PRINT FILE (1), R0$; TAB(11);
4000 NEXT L0
4010 PRINT FILE (1), R0$;
4020 CLOSE FILE (1)
4030 OPEN FILE (1,1), "$TT01"
4040 PRINT FILE (1), R0$; "CONC. ";
4070 LET T=P
4080 IF A1<3 THEN LET F3=1
4090 FOR L0=2 TO N ! PRINT CONC'NS
4100 IF C(L0)<=SCT,9)*F3 THEN GOTO 4170 ! IF <=MIN.
4110 IF C(L0)>=SCT,12)*F3 THEN GOTO 4190 ! IF >=MAX.
4120 IF C(L0)<.001 THEN GOTO 4150
4130 PRINT FILE (1), USING " ##.### ", C(L0);
4140 GOTO 4200

```

```

4150 PRINT FILE (1), USING " #.### " , C(L0)
4160 GOTO 4200
4170 PRINT FILE (1), USING " (<#.### " , S(T,9)*F3;
4180 GOTO 4200
4190 PRINT FILE (1), USING " >#.### " , S(T,12)*F3;
4200 LET T=S(T,1) : NEXT ELEMENT LINK
4204 IF L0<>15 THEN GOTO 4210
4205 IF N=15 THEN GOTO 4210
4206 PRINT FILE (1), R0* : TAB(12);
4210 NEXT L0
4240 PRINT FILE (1), R0* :
4250 PRINT FILE (1), R0*
4260 CLOSE FILE (1)
4261 PRINT "CONC'ns NOT STORED ON DISKETTE" ! REMOVE LINES 4261&4263 TO RESTORE
4262 LET F1*="SAMPLE.0", N6*(N3,N3) : FILE NAME
4263 GOTO 0930
4264 OPEN FILE (2), F1* : OPEN FOR WRITE TO DISKETTE
4266 WRITE FILE (2), I1* : WRITE SAMPLE IDENT
4268 FOR L0=2 TO N : WRITE CONC'NS
4270 WRITE FILE (2), C(L0)
4272 NEXT L0
4275 CLOSE FILE (2)
4277 LET N3=N3+1
4278 IF N3>10 THEN LET N3=0
4280 GOTO 0930
4290 REM =====
4295 REM
4300 REM ***** SUBROUTINE TO READ INT. STD. CHNL. *****
4310 REM
4320 LET N0=32768 : SET DIO BIT 0
4330 LET C2=0 : FE CHNL.=0
4340 LET G2=1 : SET GAIN
4342 LET S3=0
4345 FOR L0=1 TO 20 : 20 READS
4350 CALL 1,C2,G2,D(L) : READ INT STD. CHNL.
4392 LET S3=S3+D(L) : CUMUL. SUM
4395 NEXT L0
4397 LET D(L)=S3/20 : AVG. OF 20 READS
4398 IF F7=2 THEN GOTO 4410
4400 IF D(L)<3690/B(1) THEN GOTO 4342
4410 CALL 24,N0 : INHIBIT INPUT TO INTEGRATORS
4420 LET D(L)=D(L)*S(1)/4.1
4430 LET D(1)=D(1)+D(L) : CUMUL. TOTAL FOR INT. STD.
4435 LET X(1)=X(1)+D(L)^2
4440 RETURN
4442 REM =====
4444 REM
4445 REM ***** ROUTINE FOR STANDARDIZATION(DETERMINES B COEFF'S)
4450 REM USER FUNCTION KEY ON SCREEN MGMT. PAD (STD.)
4460 LET F7=2
4465 IF A1<>1 THEN GOTO 4477
4467 LET L2*=" CO. C, C1, S, P, SI, SI1, MN, MN1, AL, SN, CU, NI, NI1, CR, CR1,
4470 LET P=99
4472 LET N=1
4475 GOSUB 2480 : ADD ELEMENTS TO LINKED LIST
4477 MAT B=CON
4480 GOTO 0640 : CLEAR DATA, PRINT HEADING
4510 LET T1=P : CALC. B COEFF'S & STORE IN WORKING TABLE
4520 LET B(1)=900/C(1)
4530 FOR L1=2 TO N

```

```

4540 LET B(L1)=S(T1,7)/C(L1) ! CORRECT'N FACTOR FOR EACH ELEMENT
4550 LET S(T1,14)=B(L1) ! STORE FACTORS IN S ARRAY
4560 LET T1=S(T1,1) ! NEXT ELEMENT
4570 NEXT L1
4580 LET F7=1 ! STANDARDIZATION COMPLETE
4600 GOSUB 2355 ! CALC. DATE AND TIME CONSTANTS
4605 GOSUB 2280 ! PRINT HEADING
4610 OPEN FILE (1,1),"STT01" ! PRINT B COEFF'S ON PRINTER
4640 PRINT FILE (1),R0$;"COEFF'S ",R0$
4670 LET Z0=P
4675 LET N9=0
4680 PRINT FILE (1),USING " #.### "/B(1) ! FE COEFF.
4690 PRINT FILE (1),USING " #.### "/S(Z0,14)
4695 LET N9=N9+1
4700 LET Z0=S(Z0,1)
4702 IF Z0=99 THEN GOTO 4720
4705 IF N9<>13 THEN GOTO 4690
4707 PRINT FILE (1),USING " #.###",S(Z0,14)
4709 PRINT FILE (1),TAB(9)
4715 GOTO 4695
4720 PRINT FILE (1),R0$
4730 PRINT FILE (1),R0$
4731 CLOSE FILE (1)
4732 OPEN FILE (1,1),"COEFFS" ! SAVE B COEFF'S ON DISKETTE
4734 MAT WRITE FILE (1),B
4740 CLOSE FILE (1)
4745 IF A1<>1 THEN GOTO 0930
4750 LET P=99
4755 LET N=1
4757 GOTO 0485 ! ELEMENT SELECTION
4760 REM =====
4765 REM
4770 REM ***** SUBROUTINE TO PRINT ELEMENT HEADING ON VIDEO *****
4775 REM
4800 IF LEN(L0$)<=80 THEN PRINT L0$(1,LEN(L0$)-1)
4805 LET N0=159-LEN(L0$)
4807 IF N0>0 THEN LET N0=0
4810 IF LEN(L0$)>80 THEN PRINT L0$(1,79),L0$(81,LEN(L0$)+N0)
4830 IF LEN(L0$)>160 THEN PRINT L0$(161,LEN(L0$)-1)
4850 RETURN
4855 REM =====
4857 REM
4860 REM ***** ROUTINE TO RECOVER B COEFF'S *****
4865 REM
4870 OPEN FILE (1,3),"COEFFS"
4880 MAT READ FILE (1),B ! PUT COEFF'S IN B VECTOR
4890 CLOSE FILE (1)
4900 LET N0=2
4910 FOR L0=1 TO 30
4920 LET S0=S(L0,1) ! CHECK LINK
4930 IF S0=0 THEN GOTO 4960 ! NO LINK GET NEXT
4940 LET S(S0,14)=B(N0) ! STORE COEFF. IN S ARRAY
4950 LET N0=N0+1 ! INCREMENT INDEX
4960 NEXT L0
4970 RETURN
4980 REM =====
4985 REM
4990 REM ***** ROUTINE TO PRINT HEADING & EACH BURN ON PRINTER *****
4995 REM USER FUNCTION KEY CTRL S (RES) SYMBOL "8"
4997 REM

```

```

5000 GOSUB 2355 :      CALC. DATE & TIME CONSTANTS
5010 GOSUB 2280 :      PRINT HEADING
5020 FOR M=1 TO 10
5030   IF FCM> THEN GOTO 5060 : FCM> SET = DATA AVAILABLE
5040 NEXT M
5050 GOTO 0930
5060 LET L0=M*N+1
5070 OPEN FILE (1,1), "ATT01"
5080 PRINT FILE (1), USING "      (M) ", M : PRINT DATA REGION NO.
5090 FOR I=L0+1 TO L0+N-1
5100   PRINT FILE (1), USING "  ##.## ", D(I) : PRINT DATA
5110 NEXT I
5140 PRINT FILE (1), R0$
5150 CLOSE FILE (1)
5160 GOTO 5040
5170 REM =====
5180 REM
5200 REM ***** ROUTINE TO STORE CHECK SAMPLE READINGS *****
5205 REM               USER FUNCTION KEY 9(FAC) SYMBOL "9"
5207 REM
5210 IF F9 THEN GOTO 5230
5220 PRINT " * * *  AVG.  REQUIRED * * * "
5225 GOTO 0930
5230 LET T=P
5231 LET Z1=2
5232 IF T<3 THEN LET Z1=3 : CARBON IS 1ST OR 2ND
5233 IF ABS(1-SCT(7)/ACT(Z1))>=.015 THEN GOTO 0930 : IF CHANGE NOT > .015
5235 OPEN FILE (1,1), "ATT01"
5236 PRINT FILE (1), R0$
5237 PRINT FILE (1), "FACTORS " :
5240 FOR L0=2 TO N
5250   LET SCT(6)=ACT(L0)*(SCT(6)/SCT(7)) : STORE CHK. SAMPLE READING
      (UNCORRECTED)
5255   PRINT FILE (1), USING "  ##.## ", SCT(7)/SCT(6) : PRINT NEW FACTOR
5260   LET T=SCT(1) : NEXT ELEMENT
5270 NEXT L0
5275 PRINT FILE (1), R0$
5277 CLOSE FILE (1)
5290 GOTO 0930
5300 REM =====
5305 REM *****
5310 REM =====

```

```

5300 REM
5305 REM
5310 REM ***** ROUTINE TO WRITE ANALYSIS REPORT *****
5320 REM USER FUNCTION KEY 10 (PAR) SYMBOL "z"
5330 REM
5340 DIM G4$(15),G5$(8),G6$(8),G7$(8)
5350 DIM N0$(300),N1$(15),N2$(15),N3$(15),N4$(15),N5$(15)
5360 LET N0$(1,60)="D.C. BRIGGS      R.K. BUHR      A.F. CRAWLEY
      K.G. DAVIS      "
5370 LET N0$(61,120)="D.M. FEGREDO   R.F. KNIGHT   R.J. LACROIX
      J. GIANETTO     "
5380 LET N0$(121,180)="D.E. PARSONS   G.E. RUDDLE   R.G. SABOURIN
      J. HARBEC       "
5390 LET N0$(181,240)="A. PALMER     D.J. BARKLEY  R. CHARBONNEAU
      T.R. CHURCHILL  "
5400 LET N0$(241,300)="R.E. HORTON   R.A. FORTIN   L. DESJARDINS
      "
5410 LET F7=3 ! FLAG USED BY "LES" ROUTINE
5415 LET N2=N ! NO. OF ELEMENTS INCLUDING INTERNAL STD.
5420 IF LEN(L0$)<80 THEN PRINT L0$ !LIST ELEMENT NAMES ON SCREEN
5430 IF LEN(L0$)>80 THEN PRINT L0$(1,80)
5440 IF LEN(L0$)>80 THEN PRINT L0$(81,LEN(L0$))
5450 INPUT "DATA TO BE REMOVED ? NO or YES ",A2$
5460 IF LEN(A2$)=0 THEN GOTO 5540
5470 IF A2$(1,3)<>"YES" THEN GOTO 5540
5480 GOSUB 6300 ! SUBROUTINE TO SELECT DATA TO BE REMOVED
5540 INPUT "SAMPLE TYPE,REPORT NO.,PROJECT NO. ",G4$,G5$,G6$
5550 FOR J0=1 TO 17 STEP 4
5560   FOR I0=J0 TO J0+3
5570     PRINT I0;N0$((I0-1)*15+1,I0*15); ! DISPLAY LIST OF NAMES
5580   NEXT I0
5590   PRINT
5600 NEXT J0
5610 PRINT
5620 INPUT "SAMPLES FROM ",N9 ! RESPOND WITH NO. FROM DISPLAYED LIST
5630 INPUT " COPIES TO ",N8,N7,N6 ! " AS ABOVE
5640 INPUT "ANALYST ",N5 ! " AS ABOVE
5650 INPUT "No. of Samples ",N4
5660 INPUT "1ST Sample No. ",S0
5670 LET N5$=N0$((N9-1)*15+1,N9*15) ! FORM CUSTOMER NAME
5680 LET N4$=N0$((N5-1)*15+1,N5*15) ! FORM ANALYSTS NAME
5690 IF N8 THEN LET N1$=N0$((N8-1)*15+1,N8*15)
5700 IF N7 THEN LET N2$=N0$((N7-1)*15+1,N7*15)
5710 IF N6 THEN LET N3$=N0$((N6-1)*15+1,N6*15) ! SELECT 1 TO 3 NAMES
5720 OPEN FILE (1,1),"$TT01" ! PRINT ANALYSIS REPORT FORM
5730 PRINT FILE (1), TAB(46);"MINERAL SCIENCES LABORATORIES"
5740 PRINT FILE (1), TAB(43);"LABORATOIRE DES SCIENCES MINERALES"
5750 PRINT FILE (1),"<13>"
5760 PRINT FILE (1), TAB(5);"Samples/", TAB(98);"Report No."
5770 PRINT FILE (1), TAB(5);"Echantillons:",G4$, TAB(98);"No. de Rapport: ",G5$
5780 PRINT FILE (1),"<13>"
5790 PRINT FILE (1), TAB(39);"SPECTROCHEMICAL ANALYSIS/"
5800 PRINT FILE (1), TAB(5);"From/de:",N5$; TAB(39);"ANALYSE SPECTROCHIMIQUE:
      -Quantometer";
5805 PRINT FILE (1),"<10>"; TAB(98);
5810 PRINT FILE (1), TAB(98);"Date: ",SYS(1);"/";SYS(2);"/";SYS(3)
5820 PRINT FILE (1),"<13>"
5830 PRINT FILE (1), TAB(5);"Project No."
5840 PRINT FILE (1), TAB(5);"No.de Projet/";G6$
5850 PRINT FILE (1),"<13>"

```

```

5860 FOR J0=1 TO 65
5870   PRINT FILE (1); "- "
5880 NEXT J0
5890 CLOSE FILE (1)
5900 OPEN FILE (1,1); "$T701"
5910 FOR J0=1 TO 65
5920   PRINT FILE (1); "- "
5930 NEXT J0
5940 PRINT FILE (1); "<13>"
5960 PRINT FILE (1); TAB(55); "Element - 2"
5970 PRINT FILE (1); "<13>"
5980 PRINT FILE (1); TAB(30); L0$
5990 PRINT FILE (1); "Lab No."; TAB(12); "Product:Produit"
6000 PRINT FILE (1); "<13>"
6010 PRINT FILE (1); "<10>"
6020 FOR N3=1 TO N4
6030   LET F1$="SAMPLE.0".N6$(N3,N3) ! FORM FILENAME
6040   OPEN FILE (2,3); F1$
6050   READ FILE (2); I1$ ! GET SAMPLE IDENTIFICATION
6060   FOR L1=1 TO N2-1
6070     READ FILE (2); C(L1) ! GET CONCENTRATION VALUES
6080   NEXT L1
6090   CLOSE FILE (2)
6095   PRINT FILE (1); USING "####" ; S0; ! PRINT CHEM LAB SAMPLE NO.
6097   PRINT FILE (1); I1$; TAB(30); ! PRINT SAMPLE IDENTIFICATION
6100   FOR L1=1 TO N2-1
6110     FOR L2=1 TO L0-1
6120       IF F(L2)=L1 THEN GOTO 6150 ! IF TRUE LEAVE OUT VALUE
6130     NEXT L2
6140     PRINT FILE (1); USING "###.###" ; C(L1); ! PRINT CONCENTRATION VALUES
6150   NEXT L1
6160   PRINT FILE (1); "<13>"
6165   LET S0=S0+1 ! INCREMENT SAMPLE NO.
6170 NEXT N3
6180 PRINT FILE (1); "<13>"
6190 PRINT FILE (1); TAB(98); "Analyst/Analyste: "; N4$
6200 PRINT FILE (1); TAB(5); "Approved/Approuve:"
6210 PRINT FILE (1); TAB(23); "Section Head :Chef de Section"
6220 PRINT FILE (1); "<13>"
6230 PRINT FILE (1); TAB(5); "Copies to/a: 1. "; N1$; TAB(65); "4. Section Files:
Dossiers de Section"
6240 PRINT FILE (1); TAB(18); "2. "; N2$; TAB(65); "5. Laboratory Files:
Dossiers de Laboratoire"
6250 PRINT FILE (1); TAB(18); "3. "; N3$
6260 PRINT FILE (1); "<12>"
6270 CLOSE FILE (1)
6280 LET F7=1 ! RESET FLAG
6290 GOTO 0930
6300 REM ***** SUBROUTINE TO SELECT DATA TO BE REMOVED
6310 LET L0=1
6320 MAT F=ZER
6330 INPUT " POS'n No. "; A2$;
6340 IF LEN(A2$)=0 THEN GOTO 6410
6350 LET F(L0)=VAL(A2$)
6360 IF F(L0)>N1-1 THEN GOTO 6390
6370 LET L0=L0+1
6380 GOTO 6330
6390 PRINT "<13>INVALID POS'n No. DUM DUM !!!"
6400 GOTO 6330
6410 INPUT "USE THE LES FUNCTION KEY TO MODIFY HEADING "; A2$

```

6420 IF A2*(1,2)<>"(30>v" THEN GOTO 6410

6430 GOSUB 1780

6440 RETURN

6450 REM*****

6460 REM-----

APPENDIX B. Variables and Their Uses

(a) Dimensioned Vectors and Arrays

Name(Dim)	Function
A(63)	Initially for Channel and Gain no's then Data Averages (2810,2820,1582)
B(30)	Coefficients derived by Standardization (290,4540,4880)
C(25)	Initially Averages then element concentrations (1570,3170)
D(140)	Readings from spectrometer for each element and burn (1085 to 1120)
E(14)	Buffer for transfer of data from Diskette to Memory (352,390)
F(10)	Flags to indicate if data areas are available (1032,1390,1510,5030)
G(25)	Relative Standard Deviation for each element (3390)
R(25)	Ratios used for High Alloy measurements (2150)
S(30,14)	Array for data pertaining to each element (14 items/element) (390)
X(65)	Sums of Squares of data for each element (1130)
Z(65)	Mean Sum of Squares (1580)

(b) Variables

A1	To receive single digit input for alloy type selection (320)
A2	To receive single digit input for element selection (490)
C1	Index for branch on function key selection (950,970)
C2	Channel no. for reading integrators (4330)
C4	Constant for day and month calculation (2355)
D1	Sum of ratios for concentration calculation by ratio method (2110)
F0	Keyboard response for print subroutine (3535,3540,3580)
F1	Constant for day and month calculation (2380,2410)
F3	XFe derived from ratio method (2120)
F4	Flag to indicate that Si is being analyzed (2840)
F5	
F6	Flag set if both Si and Mo being analyzed (2850)
F7	Flag to indicate Standardization in progress (4460)
F8	Flag to indicate if Average has been calculated (1490)
F9	Flag to indicate if RSD has been calculated (3370)
G1	Constant for day and month calculation (2370)
G2	Gain setting for reading A/D converter (4340)
I	Loop variable (1080,1100,1430,1560,1720,1790,2140)
J	Index for element deletion routine (1820), Loop var. (3760)
J0	Loop variable (350)
J1	Loop variable (355), Index for buffer flags (1190)
J2	No. of repetitions to determine average (1510)
L	Index for storing data in D Array (1040,1085 to 1130)
L0	Loop variable and index (1055,1420,3880,3980,4090,4268,4345,4910)
L1	Index for setting gains, channel no.'s and B coeffs. (2780)
M	Loop variable (1030,1380)
N	Current no. of elements including Fe (230,1990,2720)
N0	Boolean variable for opening and closing relay (1160,1166)
N1	Constant for day and month calculation (2420)
N2	Constant for day and month calculation (2430)
N3	File index for storing concentrations on Diskette (4262)
N4	Constant used when adding elements to list (2725)
N9	Counter to determine no. of coeff's printed (4695)
P	Pointer to head of linked list (250,2710)

APPENDIX B (continued)

P1	Temporary pointer for linked list (2770,2860)
P3	Index for element removal and addition routines (1800,2490)
S0	Index for recovering B coefficients from Diskette (4920)
S3	Temporary sum for averaging multiple reads of A/D (4392)
T	Temporary pointer for linked list (1910,4070,4200)
T1	Temporary pointer for linked list (4510,4560)
W1	Response for input (452), index for line deletion (1692)
X1	Element concentration as derived by iteration (3010,3050)
Y1	Estimated concentration as derived by iteration (3020)
Z0	Derived constant for correction to Si (3250)

(c) String Variables

Name

A2\$(4)	Contains input from user function key (930)
B0\$(20)	Contains character codes for user function keys (1,50,950)
C1\$(20)	Element data filename endings (used for file search)
D1\$(21)	Three letter symbols for days of the week (160,820)
F0\$(5)	5 spaces (2800)
F1\$(13)	Element data filename (4262)
G4\$(15)	Contains values for Analysis Report (5340,5540)
G5\$(8)	Contains values for Analysis Report (5340,5540)
G6\$(8)	Contains values for Analysis Report (5340,5540)
G7\$(8)	Contains values for Analysis Report (5340,5540)
I1\$(40)	Sample identification (910,3710,4266)
L0\$(180)	Current Element symbols in format ^^xxx^^ (2800)
L2\$(170)	Current element symbols in format xxx (460,470,560,1780)
M0\$(130)	Three letter symbols for all elements used (190 to 210,1800,2490,2800)
M1\$(36)	Three letter symbols for months (220,820,2285)
N0\$(300)	Contains names for Analysis Report (5350,5360 to 5400)
N1\$(15)	Contains names for Analysis Report (5350,5690)
N2\$(15)	Contains names for Analysis Report (5350,5700)
N3\$(15)	Contains names for Analysis Report (5350,5710)
N4\$(15)	Contains names for Analysis Report (5350,5680)
N5\$(15)	Contains names for Analysis Report (5350,5670)
N6\$(10)	Codes for digits 0 to 9 inclusive (240,4262)
P0\$(4)	New page code
R0\$(4)	Carriage return code
S0\$(10)	Contains "SAMPLE#"^^^ (^=space)

APPENDIX C. ASSEMBLER LANGUAGE LISTING

```

0001 SRTB      MACRO REV 06.50      11-47:54 06/18/84
                                .TITLE SRTB
                                .ENT SRTB,BCDTB
                                .EXTD .FIX,.PLOT,.UIEX
                                .ENT AIRDR
                                .NREL
02 00000'000001 SRTB:      1
03 00001'000045'          ADCB4      /NON-INTERRUPT SINGLE CHNL. READ OF A/D
04 00002'000003'          3          /PARS. / CHNL.#,GAIN,STORE ADR
05 00003'000004'          .RDX 4
06 00004'170000'          33000000
07 00005'000010'          .RDX 8
08 00006'000012'          10.
09 00007'000027'          SETUP      /CREATES TABLE OF GAINS & CHNL. NO.'S
10 00008'000002'          2          /PARS. /# OF PTS.,ADR OF GAINS & CHNLS.
11 00009'000004'          .RDX 4
12 00010'170000'          33000000
13 00011'000010'          .RDX 8
14 00012'000014'          12.
15 00013'000436'          AIRDR      /IDENTIFIES A/D FOR INTERRUPTS
16 00014'000000'          0
17 00015'000000'          0
18 00016'000016'          14.
19 00017'000234'          ADCMD      /FLOATS AND STORES A/D READINGS
20 00018'000002'          2          /PARS. /FLAG & DATA ADRS.
21 00019'000004'          .RDX 4
22 00020'170000'          33000000
23 00021'000010'          .RDX 8
24 00022'000024'          20.
25 00023'000106'          DI0B1      /NON-INTERRUPT INPUT TO DIO BOARD
26 00024'000002'          2          /PARS. / NO. OF DATA ITEMS,STORE ADR.
27 00025'000004'          .RDX 4
28 00026'170000'          33000000
29 00027'000010'          .RDX 8
30 00028'000030'          24.
31 00029'000234'          DI0B2      /NON-INTERRUPT OUTPUT VIA DIO BOARD
32 00030'000001'          1          /PAR. /NO. TO OUTPUT(I.E. BIT STATES)
33 00031'000004'          .RDX 4
34 00032'140000'          30000000
35 00033'000010'          .RDX 8
36 00034'000007'          7
37 00035'001004'          IOPIT      /ROUTINE TO IDEF PIT & DGDAC FOR INTERRUPTS
38 00036'000000'          0
39 00037'000000'          0
40 00038'000036'          30.
41 00039'001034'          SETEM      /SET TIME INTERVALS & TRIAC SWITCHES
42 00040'000001'          1          /1 PARAMETER ADDRESS OF DATA IN BASIC
43 00041'000004'          .RDX 4
44 00042'140000'          30000000
45 00043'000010'          .RDX 8
46 00044'000041'          33.
47 00045'001064'          TRISSET      /ROUTINE TO SET TRIAC SWITCHES
48 00046'000001'          1          /DECIMAL EQUIV. OF SWITCHES TO BE SET
49 00047'000004'          .RDX 4
50 00048'140000'          30000000
51 00049'000010'          .RDX 8
52 00050'177777'          -1
53 00051'050563 ADCB4:      STA 2,ADR
54 00052'004466          JSR GFIX      /GET CHNL NO.
55 00053'065021          004 1,ADCU      /SET IN A/D
56 00054'004464          JSR GFIX      /GET GAIN

```

```

0002 SARTB
01 00051'125220      MOVZR 1,1
02 00052'125200      HOUR 1,1
03 00053'125200      HOUR 1,1
04 00054'067121      DOCS 1,ADCU      /SHIFT TO 2 HIGH ORDER BITS
05 00055'063521      SKPBZ ADCU      /SET GAIN & START CONVERTER
06 00056'000777      JMP -1          /WAIT UNTIL CONVERSION COMPLETED
07 00057'066621      DICC 1,ADCU      /GET READING
08 00060'006551      JSR @ASTOR      /FLOAT & STORE READING
09 00061'010547      ISZ ADR
10 00062'002546      JMP @ADR          /RETURN
11 00063'024535 INIT:  LDA 1,ATBL1
12 00064'044020      STA 1,20          /INIT. AUTOINCREMENT LOCATION
13 00065'024467      LDA 1,CNTR-1      /32 STORAGE LOCN'S
14 00066'044467      STA 1,CNTR
15 00067'001400      JMP 0,3
16                  /CONVERT DATA TO FLT. PT. STORE IN BASIC
17 00070'004773 CNVST: JSR INIT          /INITIALIZE
18 00071'026020      LDA 1,020          /GET BINARY DATA
19 00072'102400      SUB 0,0
20 00073'125112      MOVL# 1,1,SEC
21 00074'102000      ADC 0,0
22 00075'006002*      JSR 0,FLOT          /FLOAT DATA
23 00076'030532      LDA 2,ADR          /GET STORAGE ADR.
24 00077'041000      STA 0,0,2
25 00100'045001      STA 1,1,2          /STORE FLOATED DATA
26 00101'010527      ISZ ADR
27 00102'010526      ISZ ADR
28 00103'014452      DSZ CNTR          /FINISHED ?
29 00104'000765      JMP CNVST+1        /NO
30 00105'002452      JMP 0,RTN          /YES
31                  /DIO INPUT (ASSUMES BCD INPUT)
32 00106'050522 DIOB1: STA 2,ADR
33 00107'004435      JSR CLEAR          /CLEAR TABLE & INITIALIZE
34 00110'004424      JSR GFIX          /FIX NO. & ADVANCE POINTER
35 00111'044444      STA 1,CNTR
36 00112'044444      STA 1,CNTR1
37 00113'026515      LDA 1,0ADR
38 00114'030514      LDA 2,ADR
39 00115'151400      INC 2,2          /FORM RETURN ADR.
40 00116'050441      STA 2,RTN          /SAVE IT
41 00117'044511      STA 1,ADR          /SAVE STORAGE ADR.
42 00120'060142      NIOS DIO          /START READING
43 00121'063542 BUSY: SKPBZ DIO
44 00122'000777      JMP -1          /WAIT FOR COMPLETION
45 00123'060442      DIA 0,DIO          /GET READING
46 00124'006477      JSR @ABCD          /CONVERT TO BINARY
47 00125'046020      STA 1,020          /STORE IT
48 00126'014430      DSZ CNTR1
49 00127'000771      JMP BUSY-1        /FINISHED ?
50 00130'060242      NIOC DIO          /NO
51 00131'024467      LDA 1,ATBL1
52 00132'044020      STA 1,20          /YES, CLEAR DIO
53 00133'000736      JMP CNVST+1
54 00134'054423 GFIX: STA 3,RTN
55 00135'032473      LDA 2,0ADR
56 00136'021000      LDA 0,0,2
57 00137'025001      LDA 1,1,2
58 00140'006001*      JSR 0,FIK          /GET NO.
59 00141'063077      HALT              /FIX IT
60 00142'010466      ISZ ADR          /ADVANCE PTR.

```

```

0003 S8RT9
01 00143'002414      JMP @RTH
02 00144'054456 CLEAR: STA 3,RTH1
03 00145'004716      JSR INIT
04 00146'102400      SUB 0,0
05 00147'042020      STA 00,20      /INITIALIZE & CLEAR TABLE
06 00150'014405      DSZ CNTR
07 00151'000776      JMP -2
08 00152'004711      JSR INIT      /RE-INITIALIZE
09 00153'002447      JMP @RTH1
10 00154'000040      40
11 00155'000000 CNTR: 0
12 00156'000000 CNTR1: 0
13 00157'000000 RTH: 0
14 00150'000040 TBL1: 1,SLK 40      /ASSIGN 32, LOCH'S
15 00220'000157'ATBL1: TBL1-1
16 00221'000071'ACNUS1: CNUS1-1
17 00222'000000 RTH1: 0
18 00223'000302'ASCD: SCOTS
19      /DIO OUTPUT DRIVER NON-INTERRUPT
20 00224'050404 DIO52: STA 2,ADR
21 00225'004707      JSR GFIX      /GET NO. & FIX IT
22 00226'065042      DDA 1,DIO      /SEND TO DIO
23 00227'002401      JMP @ADR
24 00230'000000 ADR: 0
25 00231'000271'ASTOR: STOR
26 00232'000000 CNTR2: 0
27 00233'000000 ADR1: 0
28 00234'050774 ADCND: STA 2,ADR
29 00235'151400      INC 2,2
30 00236'151400      INC 2,2
31 00237'050546      STA 2,ADR2      /SAVE RETURN ADR.
32 00240'034536      LDA 3,AUCT      /GET UCT ADR.
33 00241'0325400      LDA 1,STAT 3      /GET STATUS
34 00242'125005      MOV 1,1,SNR      /BUSY ?
35 00243'000403      JMP -3      /NO
36 00244'004425      JSR STOR      /SET BASIC FLAG NON-ZERO
37 00245'002540      JMP @ADR2      /RETURN
38 00246'004423      JSR STOR      /SET BASIC FLAG=0
39 00247'020544      LDA 0,DCODE
40 00250'006017      /SYSTEM
41 00251'021010      IRMU      /REMOVE A/D FROM INTERRUPTS
42 00252'063077      HALT
43 00253'010755      ISZ ADR
44 00254'030522      LDA 2,AUCT
45 00255'025003      LDA 1,NPT 2
46 00256'044541      STA 1,DTCNT      /SET COUNTER
47 00257'025002      LDA 1,UTPTR 2
48 00260'044536      STA 1,DUPTR      /SET TABLE ADR.
49 00261'026535      LDA 1,@DUPTR      /GET A/D READING
50 00262'004407      JSR STOR      /FLOAT & STORE IN BASIC
51 00263'012745      ISZ @ADR
52 00264'012744      ISZ @ADR      /ADVANCE STORAGE PTR.
53 00265'010531      ISZ DUPTR      /ADVANCE TABLE PTR.
54 00266'014531      DSZ DTCNT      /FINISHED ?
55 00267'000772      JMP -6      /NO
56 00270'002515      JMP @ADR2      /YES, RETURN
57 00271'054434 STOR: STA 3,SAV
58 00272'102400      SUB 0,0
59 00273'125112      MOVL 1,1,%SZC      /IS NO. -UE
60 00274'102000      ADC 0,0      /YES, COMPLEMENT HIGH ORDER WORD

```

```

0004 SBRB
01 00275'0060026 JSR 0.FLOT
02 00276'032732 LDA 2.0ADR
03 00277'041000 STA 0.0.2
04 00300'045001 STA 1.1.2 ;FLOT & STORE
05 00301'002424 JMP 0SAU
06 ; BCD TO BINARY CONVERSION ROUTINE
07 ; ENTER WITH NO. TO BE CONVERTED IN AC0
08 00302'054423 BCDTB: STA 3.0AU
09 00303'176120 ADDCL 3.3 ;FORM -2
10 00304'175120 MOVCL 3.3 ;FORM -4
11 00305'054725 STA 3.0NTR2 ;COUNTER FOR 4 BCD DIGITS
12 00306'126400 SUB 1.1
13 00307'034417 LOOP: LDA 3.0MSK1 ;MASK TO GET RT. 4 BITS
14 00310'107120 ADDCL 0.0
15 00311'101100 MOVL 0.0
16 00312'101100 MOVL 0.0
17 00313'101100 MOVL 0.0 ;POS'N MSD(BCD) IN LSD
18 00314'117400 AND 0.3 ;MASK TO GET RT. 4 BITS
19 00315'101200 MOVR 0.0 ;RETRIEVE BIT IN CARRY
20 00316'131000 MOVL 1.2
21 00317'127120 ADDCL 1.1 ;AC1*4
22 00320'147120 ADDCL 2.1 ;AC1*10 I.E. (4*AC1+AC2)*2
23 00321'167000 ADD 3.1 ;ADD IN NEXT DIGIT
24 00322'010710 ISZ 0NTR2
25 00323'000764 JMP LOOP
26 00324'002401 JMP 0SAU ; EXIT WITH NO. IN AC1
27 00325'000000 SAU: 0
28 00326'000017 MSK1: 17
29 00327'050701 SETUP: STA 3.0ADR
30 00330'004604 JSR 0.FIX ;GET & FIX NO. OF DATA PTS.
31 00331'044451 STA 1.0UCT-4 ;STORE NO.
32 00332'044465 STA 1.0TCNT ;SET CNTR.
33 00333'032675 LDA 2.0ADR ;GET BASIC ADDRESS
34 00334'050451 STA 2.0ADR2 ;SAVE ADR.
35 00335'010673 ISZ 0ADR ;SET FOR RETURN
36 00336'030547 LDA 2.0AGABUF ;ADDRESS OF GAIN & CHNL. DATA BUFFER
37 00337'050444 STA 2.0UFPT
38 00340'030445 LDA 2.0ADR2 ;GET ADR. OF GAIN DATA
39 00341'021000 LDA 0.0.2 ; )
40 00342'025001 LDA 1.1.2 ; ) GET GAIN & FIX
41 00343'006001$ JSR 0.FIX ; )
42 00344'063077 HALT
43 00345'125220 MOVR 1.1
44 00346'125200 MOVR 1.1
45 00347'125200 MOVR 1.1 ; POS'N GAIN (0,1,2,3) IN BITS 0 & 1
46 00350'044434 STA 1.0TEMP1
47 00351'030434 LDA 2.0ADR2
48 00352'021100 LDA 0.100.2 ; CHNL. #'S ARE DISPLACED 64 FROM GAINS
49 00353'025101 LDA 1.101.2
50 00354'006001$ JSR 0.FIX ;GET CHNL. NO. & FIX IT
51 00355'063077 HALT
52 00356'020426 LDA 0.0TEMP1
53 00357'100000 COM 0.0
54 00360'107400 AND 0.1
55 00361'106000 ADC 0.1 ; INCLUSIVE 'OR' GAIN & CHNL. #
56 00362'046421 STA 1.00UFPT ;STORE MERGED DATA
57 00363'014434 DSZ 0TCNT ;FINISHED ?
58 00364'000402 JMP +2 ;NO
59 00365'000405 JMP FIN ;YES
60 00366'010415 ISZ 0UFPT

```

```

0005 SSBTB
01 00367'010416 ISZ ADR2
02 00370'010415 ISZ ADR2
03 00371'000747 JMP SETUP+11 /SET NEXT GAIN & CHNL. NO.
04 00372'010411 FIN: ISZ BUPT
05 00373'126400 SUB 1,1
06 00374'046407 STA 1,0BUPT
07 00375'002633 JMP @ADR
08 00376'000377'AUCT: +1 /ADR. OF USER CONTROL TABLE
09 00377'000000 0 /STATUS FLAG
10 00400'000506' ACASUF+1 /ADR. OF GAIN & CHNL ARRAY
11 00401'000547' ADBUF /ADR. OF DATA ARRAY
12 00402'000000 0 /NO. OF READINGS
13 00403'000000 BUPT: 0
14 00404'000000 TEMP1: 0
15 00405'000000 ADR2: 0
16 00406'000143 C99: 143
17 00407'000410'ADDCT: USDC /ADR. OF A/D DCT
18 00410'000420'USDC: AISAUE /INTERRUPT SAVE AREA
19 00411'000200 100 /MASK
20 00412'000457' AINTS /ADR. OF INTERRUPT ROUTINE
21 00413'000021 DCODE: ADCU /DEVICE CODE
22 00414'000000 ACTIU: 0 /STATUS FLAG
23 00415'000000 DAPTR: 0 /GAIN & CHNL ARRAY PTR.
24 00416'000000 DUPTR: 0 /INPUT DATA PTR.
25 00417'000000 DTCNT: 0 /NO. OF READINGS
26 00420'000010 AISAUE: BLK 10
27 000000 DUSR: STAT=0 /OFFSET TO STATUS FLAG
28 000001 DUSR: A PTR=1 /OFFSET TO CHNL. & GAIN ARRAY
29 000002 DUSR: UTPTR=2 /OFFSET TO INPUT DATA ARRAY
30 000003 DUSR: NPT=3 /OFFSET TO NO. OF READINGS
31 00430'050755 AICLR: STA 2 ADR2 /REMOVE A/D FROM INTERRUPTS
32 00431'020752 LDA 0 DCODE
33 00432'005017 .SYSTM
34 00433'021010 .IRMU
35 00434'063077 HALT /ERROR RETURN
36 00435'002750 JMP 0 ADR2
37 /ROUTINE TO READ SELECTED GROUP OF CHNLS. VIA A/D
38 00436'050747 AIRDR: STA 2 ADR2 /SAVE ADR. FROM BASIC
39 00437'102400 SUB 0 0
40 00440'040754 STA 0 ACTIU /CLEAR ACTIU FLAG
41 00441'020752 LDA 0 DCODE
42 00442'024745 LDA 1 ADDCT
43 00443'005017 .SYSTM
44 00444'021007 .IDEF /IDENTIFY FOR INTERRUPTS
45 00445'063077 HALT
46 00446'020746 LDA 0 ACTIU
47 00447'101004 MOV 0 0 SZP
48 00450'002735 JMP @ADR2 /RETURN IF ACTIVE
49 00451'030725 LDA 2 AUCT
50 00452'050742 STA 2 ACTIU /SET FLAG ACTIVE
51 00453'021003 LDA 0 NPT 2
52 00454'040743 STA 0 DTCNT /SET CNTR.
53 00455'021001 LDA 0 ATPTR 2
54 00456'040737 STA 0 DAPTR /SET PTR. TO GAIN TABLE
55 00457'021002 LDA 0 UTPTR 2
56 00460'040736 STA 0 DUPTR /SET STORAGE PTR.
57 00461'102000 ADC 0 0
58 00462'041000 STA 0 STAT 2
59 00463'022732 LDA 0 @DAPTR /GET 1ST CHNL. & GAIN PAIR
60 00464'063021 DDC 0 ADCU

```

```

0006 SBRIS
01 00465'061121 DOAS 0 ADCU /SET IN A/D & START
02 00466'002717 JMP @ADR2
03 /INTERRUPT SERVICE ROUTINE
04 00467'062621 AINTS: DIOC 0 ADCU /READ A/D
05 00470'042726 STA 0 @DUPTR /STORE READING
06 00471'014726 DSZ DTCHT /FINISHED ?
07 00472'000405 JMP AIMORE /NO.GET NEXT READING
08 00473'102400 SUB 0 0
09 00474'042720 STA 0 @ACTIV
10 00475'040717 STA 0 @ACTIV /FINISHED CLEAR FLAG
11 00476'001400 JMP 0 3
12 00477'019716 AIMORE: ISZ DAPTR
13 00500'010716 ISZ DUPTR /ADVANCE PTRS
14 00501'022714 LDA 0 @DAPTR /GET GAIN & CHNL.
15 00502'063021 DDC 0 ADCU
16 00503'061121 DOAS 0 ADCU /RESTART CONVERTER
17 00504'001400 JMP 0 3
18 00505'000506 ACABUF: .+1
19 00506'000041 .BLK 41
20 00547'000040 ADUF: .BLK 40
21
22
23
24 /CODE TO ATTACH THE PROGRAMMABLE INTERVAL TIMER
25 00607'000610 ADRPIT: PITDCT /ADR. OF 3 WORD TABLE
26 00610'000000 PITDCT: 0
27 00611'001000 .IS6
28 00612'000613 INPIT /ADR. OF INTERRUPT ROUTINE
29
30
31 /PROGRAMMABLE INTERVALTIMER INTERRUPT ROUTINE
32
33 00613'060243 INPIT: NIOC 43 /CLEAR TIMER
34 00614'022444 LDA 0 @TRPTR /GET TRIAC SETTINGS
35 00615'024502 LDA 1, TROIS /MODE=0, ADR.=3 (TRIAC MODULE)
36 00616'065040 DOA 1, 40 /SPECIFY CONTEXT & ADR.
37 00617'062040 DOB 0, 40 /SET TRIACS
38 00620'022444 LDA 0 @TMPTR /GET TIME INTERVAL
39 00621'101005 MOV 0, 0, SNR /MORE ?
40 00622'000003* UIEX /NO. DISMISS INTERRUPT
41 00623'010435 ISZ TRPTR /YES. ADVANCE TBL PTRS.
42 00624'010435 ISZ TMPTR
43 00625'061143 DOAS 0, 43 /START TIMER
44 00626'000003* UIEX /EXIT
45
46 /SPIT- SUBROUTINE TO START PIT SEQUENCE
47
48 00627'020410 SPIT: LDA 0, ATTBL /GET TRIAC TBL. ADR.
49 00630'040430 STA 0, TRPTR /INITIALIZE POINTER
50 00631'020405 LDA 0, AMTBL /GET TIMER TABLE ADR.
51 00632'040427 STA 0, TMPTR /INIT. POINTER
52 00633'020427 LDA 0, C5 /5 MSEC. TO CAUSE INTERRUPT
53 00634'061143 DOAS 0, 43 /START TIMER
54 00635'001400 JMP 0, 3 /RETURN
55 00636'000640 AMTBL: .+2 /TIME TABLE ADR.
56 00637'000650 ATTBL: .+11 /TRIAC TABLE ADR.
57 00640'000020 .BLK 20 /2-3 WORD TABLES IN CONTIGUOUS MEMORY
58 00660'000000 TRPTR: 0
59 00661'000000 TMPTR: 0 /2-POINTER LOCATIONS
60 00662'177773 C5: -5 /TIME CONSTANT

```

0007 SRTS

```

01
02      ;I/O SUBSYSTEM CODE FOR INTERRUPT HANDLING.
03      ;AT THE TIME OF WRITING MAR./1983, THERE WAS ONLY
04      ;ONE INTERRUPT MODULE THE GP INPUT BOARD IN SLOT 0.
05      ;HOWEVER THE INTERRUPT ROUTINE IS GENERAL SO
06      ;IT WILL BE EASY TO ADD OTHER MODULE ROUTINES.
07      ;THE ROUTINE IGNORES CONTROLLER INTERRUPTS
08      ;AND RESTARTS THE CONTROLLER.
09 00663'000664'ADGDCI: DGDCT      ;ADR. OF DGDAC DCT
10 00664'000665' DGDCT: 0          ; 3-WORD TABLE
11 00665'002000      185          ;INTERRUPT MASK BIT
12 00666'000667'      DGINT      ;ADR. OF INTERRUPT ROUTINE
13
14
15 00667'050432 DGINT: STA 2,DCAC2      ;SAVE AC2 & AC3
16 00670'054432      STA 3,DCAC3
17 00671'034434      LDA 3,DGTBL      ;ADR. OF TABLE OF ROUTINES
18 00672'060740      DIAP 0,40      ;IDENTIFY INTERRUPT
19 00673'040425      STA 0,STATE      ;SAVE STATUS
20 00674'060640      DIAC 0,40      ;READ STATUS & CLEAR
21 00675'024426      LDA 1,MSK11     ;BIT 11 MASK CONTROL BIT
22 00676'107404      AND 0,1,SR      ;IS IT CONTROLLER INT.?
23 00677'000405      JMP CNTRLR      ;YES
24 00700'030424      LDA 2,ADRMASK    ;NO, GET ADR. MASK
25 00701'113400      AND 0,2         ;FORM MODULE ADR.
26 00702'173000      ADD 3,2         ;FORM TABLE ADR.
27 00703'007000      JSR 00,2
28 00704'020526 CNTRLR: LDA 0,STAT      ;CM=CP=1
29 00705'061140      DOAS 0,40      ;RESTART SUBSYSTEM
30 00706'020412      LDA 0,STATE      ;GET STATUS AT INTERRUPT
31 00707'034420      LDA 1,STMSK      ;CM & ADR. MASK
32 00710'107400      AND 0,1         ;FORM CM & ADR.
33 00711'125100      MOVL 1,1
34 00712'125240      MOVOR 1,1      ;SET BIT 0
35 00713'065040      DDA 1,40      ;RESTORE PREVIOUS STATUS
36 00714'030405      LDA 2,DCAC2
37 00715'034405      LDA 3,DCAC3      ;RESTORE AC2 & AC3
38 00716'000003*      JUIEX          ;INTERRUPT EXIT
39 00717'000003 TROIS: 000003
40 00720'000000 STATE: 0
41 00721'000000 DCAC2: 0
42 00722'000000 DCAC3: 0
43 00723'000020 MSK11: 000020
44 00724'000017 ADRMASK: 000017
45 00725'000726 DGTBL: 1,1
46 00726'000730 GPINT: GPINT      ;GP INPUT ROUTINE
47 00727'000037 STMSK: 000037
48
49      ;GENERAL PURPOSE INPUT BOARD ROUTINE.
50      ;ENTER WITH CONTROLLER IN MODULE MODE.
51
52 00730'054440 GPINT: STA 3,GPSPV
53 00731'061440      DIB 0,40      ;READ INPUT DATA
54 00732'034414      LDA 3,GPTBL      ;GET TABLE ADR.
55 00733'024434      LDA 1,PREV      ;GET PREVIOUS SWITCH SETTINGS
56 00734'131000      MOV 1,2        ;EXCLUSIVE OR OF AC0 & AC1
57 00735'113520      ANDZL 0,2      ;RESULT IN AC1
58 00736'107000      ADD 0,1        ;BITS WHICH CHANGED WILL BE SET
59 00737'146400      SUB 2,1        ;SAVE CURRENT CONDITION
60 00740'040427      STA 0,PREV

```

```

0000 SERTB
01 00741'125120 MORE:  MOUZI 1,1      ;GET M.S. BIT INTO CARRY
02 00742'175403      INC 3,3,SNC  ;CARRY SET =BIT SET ADVANCE PTR.
03 00743'000776      JMP -2      ;GET NEXT BIT
04 00744'044425      STA 1,FLAG0  ;IF FLAG < 0, RETURN TO MORE
05 00745'003400      JMP 00,3    ;JUMP TO SELECTED ROUTINE
06
07 00746'000746'GPTBL:
08 00747'000772'      BIT0        ;ROUTINE FOR BIT 0
09 00750'000017      .BLK 17      ;SPACE FOR 15 ADDRESSES
10 00757'000000 PREU:  0
11 00770'000000 GPSAU: 0
12 00771'000000 FLAG0: 0
13 00772'030411 BIT0:  LDA 2,MSK0  ;GET BIT 0 MASK
14 00773'113405      AND 0,2,SNC  ;IS BIT 0 SET
15 00774'000402      JMP +2      ;NO, IT WAS 100 TRANSITION, IGNORE
16 00775'004632      JSR SPIT    ;YES START INTERVAL TIMER
17 00776'125004      MOU 1,1,SZR  ;ANOTHER INTERRUPT ?
18 00777'000742      JMP MORE    ;YES
19 01000'102400      SUB 0,0
20 01001'061140      DOAS 0,40    ;SET MODULE 0 BUSY
21 01002'002766      JMP 0GPSAU  ;RETURN
22 01003'100000 MSK0:  100000
23
24      ;CALL 3 ROUTINE TO IDENTIFY DGDAC & INT. TIMER
25      ;FOR INTERRUPTS & INITIALIZE CNTRLR & INPUT MODULES
26      ;NO PARAMETERS REQUIRED
27
28
29 01004'050425 IOPIT:  STA 2,SAU0  ;ERROR CODES USE AC2
30 01005'020423      LDA 0,DGIOS  ;DEVICE CODE 40
31 01006'024655      LDA 1,ADGDCT ;ADR. OF DCT
32 01007'006017      .SYSTM
33 01010'021007      .IDEF
34 01011'063077      HALT
35 01012'020415      LDA 0,PITCOD  ;IDENT. PIT
36 01013'024413      LDA 1,PITADR
37 01014'006017      .SYSTM
38 01015'021007      .IDEF
39 01016'063077      HALT
40 01017'020413      LDA 0,STRT    ;CN=CP=1
41 01020'024413      LDA 1,CRMSK  ;CNTRLR SUB MASK
42 01021'061140      DOAS 0,40    ;SPEC. CONTEXT, SET CNTRLR BUSY
43 01022'067040      DOC 1,40     ;SET SUBMASK
44 01023'102400      SUB 0,0
45 01024'061140      DOAS 0,40    ;SET MODULE 0 BUSY
46 01025'002404      JMP 0SAU0
47 01026'000610'PITADR: PITDCT
48 01027'000043 PITCOD: 43
49 01030'000040 DGIOS:  40
50 01031'000000 SAU0:   0
51 01032'100037 STRT:   100037   ;CN=CP=1
52 01033'007440 CRMSK:  007440
53
54      ;CALL 30 SETEM ROUTINE TO SET UP TABLE FOR TRIACS
55      ;AND TIME INTERVALS
56      ;1 PARAMETER ADDRESS OF TABLE, TRIAC SETTINGS DISPLACED
57      ;8(16) FROM TIME INTERVAL ADRS.
58 01034'050775 SETEM:  STA 2,SAU0
59 01035'032774      LDA 2,0SAU0  ;GET TRIAC TABLE ADR.
60 01036'050423      STA 2,ADR3   ;SAVE BASIC TBL ADR.

```

```

0000 SERT2
01 01037'020424 LDA 0,C15
02 01040'040423 STA 0,CNTR3
03 01041'022417 LDA 0,BPTR3
04 01042'040617 STA 0,TNPT3
05 01043'030415 LOOP1: LDA 2,ADR3
06 01044'021000 LDA 0,0.2
07 01045'025001 LDA 1,1.2
08 01046'006001* JSR 0,FIX
09 01047'063077 HALT
10 01050'046611 STA 1,BTNPTR
11 01051'010610 ISZ TNPT3
12 01052'010407 ISZ ADR3
13 01053'010406 ISZ ADR3
14 01054'014406 DSE CNTR3
15 01055'000766 JMP LOOP1
16 01056'010753 ISZ SAU0
17 01057'002752 JMP BSAU0
18 01060'000636 PTR3: CNTR3
19 01061'000000 ADR3: 0
20 01062'000000 CNTR3: 0
21 01063'000020 C15: 20
22
23 ;CALL 33-----A ROUTINE TO SET TRIAC SWITCHES.
24 ;NEEDS ONE PARAMETER,THE DEC EQUIV OF THE SWITCHES
25 ;WHICH ARE TO BE SET.
26
27 01064'050775 TRIS2: STA 2,ADR3
28 01065'060440 DTA 0,40 ;READ STATUS
29 01066'040743 STA 0,SAU0 ;SAU0 STATUS
30 01067'020630 LDA 0,TRDIS ;CN=0 ADR.=3
31 01070'061040 DDB 0,40 ;SET ADR. & CONTEXT
32 01071'035000 LDA 3,0.2
33 01072'021400 LDA 0,0.3
34 01073'025401 LDA 1,1.3 ;GET DECIMAL NO.
35 01074'006001* JSR 0,FIX ;FIX IT
36 01075'063077 HALT
37 01076'066040 DDB 1,40 ;SET TRIACS
38 01077'020732 LDA 0,SAU0 ;GET STATUS
39 01100'024637 LDA 1,STMSK ;CN & ADR. MASK
40 01101'097400 AND 0,1
41 01102'125100 MOVL 1,1
42 01103'125240 MOVL 1,1 ;SET BIT 0
43 01104'005040 DDB 1,40 ;RESTORE SUBSYSTEM
44 01105'010754 ISZ ADR3
45 01106'002753 JMP BADR3 ;RETURN
46 .END

```

**00000 TOTAL ERRORS, 00000 PASS 1 ERRORS